Contributions to the Computational Treatment of Non-Literal Language

Omid Rohanian

A thesis submitted in partial fulfilment of the requirements of the University of Wolverhampton for the degree of Doctor of Philosophy

2020

This work or any part thereof has not previously been presented in any form to the University or to any other body whether for the purposes of assessment, publication or for any other purpose (unless otherwise indicated). Save for any express acknowledgements, references and/or bibliographies cited in the work, I confirm that the intellectual content of the work is the result of my own efforts and of no other person.

Signature: . . . . . .

Date: . . . . . .

Abstract

Non-literal language concerns the deliberate use of language in such a way that meaning cannot be inferred through a mere literal interpretation. In this thesis, three different forms of this phenomenon are studied; namely, irony, non-compositional Multiword Expressions (MWEs), and metaphor.

We start by developing models to identify ironic comments in the context of the social micro-blogging website Twitter. In these experiments, we proposed a new way to extract features based on a study of their spatial structure. The proposed model is shown to perform competitively on a standard Twitter dataset.

Next, we extensively study MWEs, which are the central point of focus in this work. We start by framing the task of MWE identification as sequence labelling and devise experiments to see the effect of eye-tracking data in capturing formulaic MWEs using structured prediction.

We also develop a novel neural architecture to specifically address the issue of discontinuous MWEs using a combination of Graph Convolutional Neural Networks (GCNs) and self-attention. The proposed model is subsequently tested on several languages where it is shown to outperform the state-of-the-art in overall criteria and also in capturing gappy MWEs.

In the final part of the thesis, we look at metaphor and its interaction with

verbal MWEs. In a series of experiments, we propose a hybrid BERT-based model augmented with a novel variation of GCN where we perform classification on two standard metaphor datasets using information from MWEs. This model which performs at the same level with state-of-the-art is, to the best of our knowledge, the first MWE-aware metaphor identification system paving the way for further experimentation on the interaction of different types of figurative language.

Acknowledgements

This dissertation would not have been possible without the technical and emotional support of several wonderful people who helped me throughout my PhD studies.

I would first like to express my profound gratitude to my director of studies Prof Ruslan Mitkov, from whom I have received invaluable advice and guidance throughout the course of my studies. His insight and knowledge helped shape the overall direction of the thesis and I am very grateful for his continued encouragement and help in spite of his busy schedule.

I would like to extend my sincere thanks to my first supervisor Dr Le An Ha for his dedicated guidance, kindness, and support throughout my studies. His scientific and technical prowess immensely benefited my work and our regular meetings and conversations were always constructive and positive. His vision significantly influenced both the direction of the research and the specifics of the technologies used in my experiments. I also owe a special word of thanks to my second supervisor, Dr Victoria Yaneva. She was patient and supportive and helped me freely explore creative ideas. She encouraged me to experiment with behavioural data which was instrumental in shaping the early phase of my research, and provided positive feedback and comments on my research throughout the process.

I was lucky to collaborate with Dr Shiva Taslimipoor on several publications during this time to whom I owe a debt of gratitude. She is an incredibly talented and hardworking colleague and also a wonderful human being. Her work ethic, attention to detail, and high standards in research were truly inspirational. These

collaborations were remarkably educational, and formative, and helped me explore new areas in machine learning. I would also like to thank Dr Marek Rei for his valuable contributions to the metaphor processing project that helped me connect different parts of my research and bring it to a conclusion.

I had the pleasure of meeting and forming friendships with many bright people from the Research Institute in Information and Language Processing during this time. I am in particular grateful to Prof Mike Thelwall and Dr. Kayvan Kousha who provided helpful feedback and advice on my work. I want to also thank Dr Samaneh Kouchaki, Dr Afsaneh Fazly, Dr Richard Evans, and Dr Sara Moze with whom I collaborated on several projects. It has been an honour to have Professor Aline Villavicencio and Dr Paul Wilson as my examiners. I appreciate their valuable comments and the time they spent to read and evaluate the thesis.

Last but not least, I want to thank my family members for their emotional support, understanding, and patience during this time.

Contents

List of Tables

List of Figures

Glossary

Association for Computational Linguistics The international scientific society for academics and professionals working in the area of Natural Language Processing and Computational Linguistics.

Bidirectional Long-Short Term Memory A variation of the Long-Short Term Memory (LSTM) in which two LSTM layers process the sequence in opposite directions.

Computational Linguistics An interdisciplinary science at the intersection of Computer Science and Linguistics in which computational approaches are applied to linguistic questions. In recent years, this term is increasingly conflated with its engineering counterpart, namely, Natural Language Processing (NLP).

Convolutional Neural Network A type of Neural Network that usually consists of convolutional and pooling layers.

Conditional Random Field A statistical modelling method used in structured prediction.

Graph Convolutional Network A variation of CNN where convolution is applied on the nodes of a graph.

Inside-Outside-Beginning An annotation scheme used in sequence labelling tasks where labels `I`, `O`, and `B` signify inside, outside, and beginning of a sequence.

Logistic Regression A popular statistical method used in classification that utilises the logistic function to assign probabilities to possible classes.

Long-Short Term Memory A type of RNN heavily used in NLP/CL. It consists of a cell, along with input, output, and forget gates.

Light Verb Construction A type of MWE that consists of a verb with little semantic content, followed by a noun or a compound. e.g. *make the bed*, or *give a lecture*.

Machine Learning A field of engineering that combines knowledge of statistics and computer science to develop models that can automatically detect patterns from training data and perform tasks on unseen data without explicit programming.

Masked Language Modelling One of the two unsupervised tasks (along with Next Sentence Prediction) used in the pre-training stage of BERT, in which language modelling is performed on an input where some of the tokens are randomly masked.

Multiword Expression An expression that is comprised of at least two words.

Named Entity Recognition The task of automatic identification of named entities (i.e. real-world entities such as places, organisations, persons etc) in text.

Natural Language Processing A field of engineering in which the focus is to develop models capable of understanding, generating, and manipulating human language. It is sometimes used interchangeably with Computational Linguistics, with a focus on practical engineering rather than theoretical studies.

Neural Network A computational model used in machine learning, originally inspired by the human nervous system.

Next Sentence Prediction One of the two unsupervised tasks used in the pre-training stage of BERT (the other being MLM), where the model predicts the next sentence given an input sentence.

Part of Speech The grammatical category a word is assigned to in a sentence (e.g. noun, verb, adjective, or adverb).

Recurrent Neural Network A type of neural network that allows loops with the aim to preserve temporal information.

Singular Value Decomposition A matrix factorization method commonly used in machine learning.

Support Vector Machine A popular supervised learning method based

on a separating hyperplane which can be applied to both classification and regression.

Verb-Particle Construction Sometimes called phrasal verbs, they are formed by a head verb followed by an adverb or a preposition (e.g. *ask around*, or *break down*).

List of Acronyms

ACL Association for Computational Linguistics

Bi-LSTM Bidirectional Long-Short Term Memory

CL Computational Linguistics

CNN Convolutional Neural Network

CRF Conditional Random Field

GCN Graph Convolutional Network

IOB Inside-Outside-Beginning

LR Logistic Regression

LSTM Long-Short Term Memory

LVC Light Verb Construction

ML Machine Learning

MLM Masked Language Modelling

MWE Multiword Expression

NER Named Entity Recognition

NLP Natural Language Processing

NN Neural Network

NSP Next Sentence Prediction

POS Part of Speech

RNN Recurrent Neural Network

SVD Singular Value Decomposition

SVM Support Vector Machine

VPC Verb-Particle Construction

List of Publications

Parts of this thesis have appeared in the following peer-reviewed publications:

- Rohanian, O., Taslimipoor, S., Yaneva, V. and L. A. Ha. (2017), Using Gaze Data to Predict Multiword Expressions. *In Proceedings of the 11th Conference on Advances in Natural Language Processing (RANLP 2017), Varna, Bulgaria.* [Best Paper Award]
- Rohanian, O., Taslimipoor, S., Evans, R., and Mitkov, R. (2018). WLV at SemEval-2018 Task 3: Dissecting Tweets in Search of Irony. *In Proceedings of The 12th International Workshop on Semantic Evaluation (pp. 553-559)*
- Rohanian, O., Taslimipoor, S., Kouchaki, S., Ha, L.A., and Mitkov, R. (2018) Bridging the Gap: Attending to Discontinuity in Identification of Multiword Expressions. *In Proceedings of the 17th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2019)*
- Rohanian, O., Rei, M., Taslimipoor, S., and Ha, L.A. (2020) Verbal Multiword Expressions for Identification of Metaphor. *In Proceedings of the 58th annual meeting of the Association for Computational Linguistics (ACL 2020)*

Chapter 1

Introduction

The elusive nature of meaning in language has baffled many linguists, philosophers of language, and intellectuals throughout history. For a variety of reasons, what is expressed can differ drastically from what is implied. Social, political, religious and personal considerations preclude humans from expressing themselves with direct candour. Many a time, as a result, meaning gets shrouded behind layers of irony, humour, metaphor, rhetoric and other creative exploitations. Orators, politicians, and comedians are examples of professionals who use rhetorical devices and figures of speech in order to entertain, influence or manipulate their target audiences.

It should be noted, however, that difficulties in comprehension are not always necessarily a result of deliberate planning on the part of the speaker. The common culprits are inherent difficulty of the content (i.e. high cognitive and knowledge demand) (Scheuneman et al., 1991), lack of coherence in organisation (Bamberg, 1983), poor exposition (Olah and Carter, 2017), inarticulacy, language impediments or a combination of these (Norbury and Bishop, 2003).

Non-literal (also referred to as figurative) language, is a phenomenon

wherein the speaker *deliberately* deviates from the conventionally understood semantics of words, in such a way that comprehension is not possible through a mere literal interpretation (Sikos et al., 2008). Sometimes this shift in meaning involves invoking an image (hence the name figurative) as a way to draw attention to an essentially dissimilar thing from what is being said (Lazar, 1996). For comprehension to occur, context needs to be taken into account (Li and Sporleder, 2010). Some examples of non-literal language are irony and sarcasm, pun, metaphor, and idiom.

In some cases, as in metaphors, there is a subtle connection between the original (i.e. literal) utterance and the figurative meaning derived from it. Uncovering meaning might require the reader to discover this link, leading to the assumption that decoding figurative language requires increased cognitive effort (Glucksberg et al., 2001). For instance, in the sentence "London is a melting pot", the original meaning refers to a container in which metals are melted and mixed. The metaphorical sense refers to a place where diverse communities of people with distinct cultures blend and assimilate into a common hybrid culture. The metaphor borrows the element of admixture from the literal sense.

As for puns, the polysemous nature of certain words is utilised for humorous effect. In the sentence "It's called the American Dream because you have to be asleep to believe it" (Carlin, 2005) the double meaning of dream as "images, thoughts, and sensations experienced during sleep" and "a cherished aspiration and ambition" is exploited to provide satirical social commentary.

Non-literal language can involve a shift in the original compositional meaning, to the extent that the intended message is in direct contrast to what is being uttered. In such cases, there is some displaced element that signals the presence of a phenomenon such as sarcasm. In the context of social media, this could translate to the presence of a hashtag expressing an opposing emotion in relation to the text. As an example of sarcasm, consider the following two sentences:

(1) Your face is not suitable for television.

(2) You have a face that is perfect for the radio.

The two sentences have a similar meaning and are comparable in terms of overall syntactic and lexical difficulty. The first one is direct and context-free, and its meaning is easily derived compositionally. The second one, however, requires understanding the incongruity between 'radio', a device that engages a human's auditory sense, with 'face' which regards visual appearance. The listener is notified of the element of sarcasm through a shift in modality (Chandler, 2007). These sentences are usually read with initial confusion and an ensuing "aha moment", and despite the use of the positive word 'perfect', the sentence is in fact conveying a negative opinion about the person's appearance.

Another ubiquitous phenomenon that relates to the idea of non-compositionality is multiword expressions (MWEs). MWEs, which include idioms, are multi-

word units that can be semantically opaque (i.e. the semantics of constituents do not add up compositionally to form the overall meaning). Consider examples like 'break a leg', 'miss the boat', and 'warp one's head around something'. All of these expressions are non-literal. The lines between figuration and MWEs can be blurry (Baldwin and Kim, 2010). For instance, expressions like 'bull market' (where ferociousness and belligerence of a wild animal is compared to the rising prices in a market) and 'cut to the chase' (a reference to action films where the chase scenes are the most exciting to the viewer) encode metaphorical meanings.

Drawing on the latest developments in representation learning, machine learning, and natural language processing, in the present research we seek to develop models capable of identifying different types of non-literal language with an eye to discovering the differences and commonalities that should be considered in the computational treatment of each type.

To have a manageable scope, we will focus on certain types of non-literal language and aim at developing models capable of classification and in-context tagging. The main focus is on non-compositional multiword expressions. There will also be experiments with irony/sarcasm [1] and metaphor detection. It should be noted that conflation of lexicalised (e.g. idioms) and non-lexicalised (e.g. irony, metaphor, etc.) figurative language under the rubric of non-literality can potentially be problematic since in the former we

[1] In this research, we take irony and sarcasm to be of the same category. In some sources, sarcasm is considered to be a subtype of irony.

deal with conventionalised expressions that show a higher degree of fixedness as opposed to the latter where there is much higher variability in the possible number of emerging linguistic patterns. However, as Do Dinh et al. (2018) have argued, the distinction between idiomaticity and metaphoricity is not a clear one, as non-literal phenomena are subjectively delineated in the literature, and there is no consensus on what non-literality means. This is reflected in the differences in the annotated datasets and the overall sparseness of the resources in this area. Besides, in many cases, the feature sets used to model various types of non-literal language bear significant overlap and similarities with one another. If information gained in the process of learning one of these phenomena is potentially transferable for application into another one, lack of annotated resources can be largely overcome by using standard methods and resources.

The research questions in this work can be summarised as the following:

Research Question 1. To what extent can state-of-the-art models identify non-literal language use in text?

The task of non-literal language identification can be modelled as either classification where an entire phrase or sentence is tagged or sequence labelling where each token is assigned a tag. Depending on how each task is modelled and given specific constraints in each case, we will devise methods that surpass or rival state-of-the-art and examine the factors that contribute the most to the identification of non-literal language.

Research Question 2. What are the differences and similarities

in modelling different forms of non-literal language?

Multiword Expressions, Irony, and Metaphor are three examples of non-literal language. In spite of this commonality, they exhibit differences that might require different computational approaches for their identification. We will examine the features and methods that can work across the board and the areas where different routes need to be explored.

Research Question 3. To what extent can representation learning improve identification of non-literal text?

There are numerous different approaches available in the literature to learn semantic representations of words and phrases. Some like `word2vec` or GLoVe have become the de facto standards in most NLP applications while contextualised approaches are quickly making inroads into their usage share. We will make use of different techniques in representation learning and compare the results in the tasks of interest.

Research Question 4. Can features from different phenomena help identify one particular kind of non-literal language?

Studies like Bingel and Søgaard (2017) suggest that identification of MWEs is improved through joint training with related semantic and syntactic tasks. This provides the motivation to look into this matter further and conduct similar experiments in our area of concern. We would be interested to see whether it is possible to extract features and information from one task and improve performance in identification of another type of non-literal language.

The main contributions of this work are listed below:

- We developed the first MWE tagging model that integrates behavioural features from eye-tracking data with linguistic information in a structured prediction task. We also analysed the differences in the efficacy of these features in L1 and L2 speakers of English. This work demonstrated how the integration of behavioural features could improve the identification of MWEs.

- As part of the eye-tracking experiments, we released annotations for verbal MWEs and made them publicly available, so other researchers can further work on similar data and reproduce and analyse our results.

- We devised a feature-rich method to analyse and identify irony in the context of the social media platform Twitter. We introduced a novel way to represent the information contained in an ironic message based on the spatial breakdown of tweets. Our models and the combination of features we used fared very well in a competitive shared task (ranking 3rd out of 44 teams), further proving the relevance of our feature engineering. We analysed how well these models performed in identifying different forms of irony.

- In a collaborative project, we extensively studied the issue of discontinuity in the context of identifying verbal MWEs and showed how sequential models are often unable to capture them in running text.

To alleviate this problem, we devised a series of deep learning models that specifically addressed this challenging issue. We developed a novel neural architecture, integrating Graph Convolutional Networks (GCNs) and Self-attention into an MWE tagging model. This linguistically interpretable language-agnostic model which used contextualised representations was tried on a standard multilingual dataset, and set a new state-of-the-art across several different languages. The code and analyses used in this work were made publicly available.

- We looked at the interplay of verbal metaphors and the closely related phenomenon of verbal metaphor in English. Through a series of experiments relying on the latest developments in contextualised representation learning, we developed models that are reliant on information from MWEs in order to identify metaphorical language. This is, to the best of our knowledge, the first MWE-aware metaphor detection model and paves the way for further exploration in this area.

The structure of the final thesis will be as follows:

In Chapter 2, each non-literal phenomenon under question, namely multiword expressions, ironic language and metaphor will be separately introduced. We provide an overview of their definitions, and explore their similarities and differences. In the second part of the chapter, we will introduce most of the main computational tools and methods used in the design of the experiments that will follow in the upcoming chapters.

In Chapter 3, we focus on irony as a figure of speech and an example of non-literality, and explore its computational treatment. In this chapter, irony is modelled with an eye to the antithetical nature of its constituents. Through extensive feature engineering, we develop a model capable of detecting ironic instances on social media and test the models on a standard dataset.

In Chapter 4, we will start by exploring MWEs while framing the task as sequence labelling. In the experiments done in this chapter, we will have a look at the influence of behavioural data to help a tagger identify MWEs in context. The chapter also includes a comparison between the predictive power of early and laze gaze features in the identification of MWEs and how the effect of these features might differ in the case of L1 and L2 speakers.

In Chapter 5, the focus is on the identification of multiword expressions for a variety of different languages while approaching it as a sequence labelling problem. We set up different experiments and devise various models to develop robust language-independent models that reach or surpass state-of-the-art. We will particularly focus on the issue of discontinuity in MWEs and introduce a novel neural architecture specifically designed to alleviate this issue.

In Chapter 6, we take a look at metaphor as another example of figurative language. Experimenting on established datasets in this area, we develop models to identify metaphorical instances using the latest development in deep learning. In a series of experiments, we will have a look at the effect of MWE and syntactic features on metaphor classification models. We will also

build on the insights from Chapter 5, and devise a modified version of the model used for MWE identification that could integrate different syntactic and semantic information in the task of metaphor detection.

Finally, in Chapter7, we conclude by comparing the considered phenomena, and enumerate the insights gained in the study of each phenomenon. In this chapter, we will have a look at the challenges faced in tackling these tasks and the achievements of the research study. In the end, we will describe possible future directions.

Chapter 2

Definitions and Theoretical Foundations

Overview. In this chapter, we will introduce the main theoretical assumptions underpinning the rest of the thesis including definitions of the terms used, and the main machine learning components that are repeatedly employed in the design of the experiments. In Section 2.1, we will first have a look at the definition of non-literal language and the reasons it appears so often in human communication. In 2.2, we will briefly introduce the three cases of non-literal language that we have chosen to study, along with a description of their similarities and differences. Finally, in Section 2.3, the nuts and bolts of the learning algorithms are introduced. This includes neural and non-neural machine learning models that have been used in the experiments of the upcoming chapters.

2.1 Non-literal Language

In this work, **non-literal language** refers to the linguistic phenomenon wherein word meanings deviate from their conventional semantics to varying degrees (Glucksberg et al., 2001; Montgomery et al., 2007). The term 'conventional' requires further explanation. For a single word, **conventional**

meaning corresponds to its entry in a dictionary and widely accepted to be the default range of meaning by the speakers of a language (Velasco, 2007). Given a phrase or a sentence, it is meant to be the logical form representation obtained by combining the lexical meanings of individual components without any further contextual assumptions (Giora, 1997).

There are different possible incentives behind this deliberate departure from conventionality. Some types of non-literal language like idioms and set phrases are easily recognised in everyday text and conversation. Others, like novel metaphors, puns, and ironic remarks might be harder to detect as they belong to creative language use and depend more heavily on the particular context in which the speaker produces them. Given all these differences, however, we can list a number of major reasons why non-literal language is created and employed(Gerrig and Gibbs Jr, 1988; Roberts and Kreuz, 1994):

- Indirectness: Sometimes people prefer to express their intent indirectly due to various reasons including personal considerations, playfulness, signalling intelligence and eloquence, elevation of social status, social taboos, limits on freedom of expression, and religious or political reasons.

- Expressing new concepts: New concepts appear every day in human language, and there is a constant need to expand the lexicon to explain the new phenomena appearing in the world. To coin new words and phrases, sometimes there is a need to assign new meaning to a word (or

a combination of words) in ways that a purely compositional semantic analysis would fail to account for.

- Conciseness: Poets use creative language to convey complex ideas and invoke metaphorical imagery in a succinct way. Writers, sometimes, deliberately use only a few words in order to get their message across in a powerful way. In ordinary conversation, people tend to use fewer words to refer to a general concept that would otherwise require direct and even lengthy explanation (e.g., a metaphorical proverb that emphasises a particular moral virtue without the need to discuss the matter with all the details). This results in phrases and words that refer to a bigger picture, hardly understood by a literal interpretation of the components.

- Rhetorical effect: Rhetorical devices are used to evoke an emotional response and enhance the persuasiveness of the expressed message. Orators, politicians, public speakers, and educators all use non-literal language to stimulate the audience in order to increase the effectiveness of their message, bringing more interest and attention to a subject.

As defined here, non-literal language is an umbrella term that can encompass a wide range of linguistic phenomena. Examples include simile, pun, metaphor, hyperbole, idiom, irony, oxymoron, personification, allusion, and paradox. Since semantic change plays a prominent role in most types of non-literal language, some texts refer to it as **trope** (Baldick, 2001), from

the Greek word *tropos*, which means "a turn, a change" (Oxford Learner's Dictionaries, 2020d). This points to the movement from the literal to the intended sense by the speaker.

Figurative language is another common term to refer to a similar idea, and each of the examples mentioned above is sometimes called a figure of speech. In this context, the emphasis is on rhetorical effect (Vervaeke and Kennedy, 1996) and playful intent (Ritchie and Dyhouse, 2008), which is the reason the speaker employs this linguistic device. Etymologically, the word figurative is derived from "figure, image" (Oxford Learner's Dictionaries, 2020a), implying that the departure from conventional semantics usually involves invoking some form of imagery.

Rhetorical devices may go beyond mere exploitation of word meaning and can also alter the structure of the sentence to achieve a desired effect. If the figure of speech relies on word meanings it is categorised as a trope, but if it involves alteration of the ordinary sequence of words or the overall structure of the sentence then it is considered to be a **scheme** which does not necessarily change the overall semantics of an utterance (Baldick, 2001). Examples of scheme include alliteration (use of words with similar sounds and letters), polyptoton (repetition of words from the same root), and antanaclasis (repetition of a word in two different senses).

The focus in this thesis is specifically on several cases of non-literal language (which overlap with the concept of trope) and not on rhetorical devices in general. However, non-literal language can sometimes involve both of these

phenomena at the same time, as in example (1):

(1) Banning Persian is 'Farsical'.

The word *farcical* is deliberately misspelt to carve out the word *Farsi* which is the endonymic counterpart to *Persian*. Therefore the humorous non-literal sentence is dependent on the polyptotonic interplay of these two related words bringing the scope of context beyond a single word. We argue that, regardless of the stylistic differences among various forms of non-literal language, a context-aware machine learning architecture would be able to take into account processes involved in what ultimately signals the figurative nature of an utterance.

In many cases, there is a traceable link between the constructed imagery in a non-literal utterance and the message intended by the speaker. For instance, in the expression *meet one's maker*, which means 'to die', the reference is to the religious belief of resurrection. However, this connection is much more opaque in the expressions *kick the bucket* or *buy the farm* which have the same meaning, but the imagery they invoke does not help construct their sense. In such cases, a purely compositional approach would lead to failure.

2.2 Three Cases of Non-literal Language Studied in This Thesis

To have a manageable scope for this thesis, we will focus on three types of non-literal language: metaphor, irony, and Multiword Expressions. Below we will have a look at their respective definitions and general properties.

2.2.1 Metaphor

metaphor comes from the Greek root metaphora, meaning to 'transfer'. The Oxford Dictionary defines metaphor as the following(Oxford Learner's Dictionaries, 2020c):

"A word or phrase used to describe somebody/something else, in a way that is different from its normal use, in order to show that the two things have the same qualities and to make the description more powerful, for example She has a heart of stone."

Metaphor has been described as the 'central trope' (Glucksberg et al., 2001) denoting its importance among rhetorical devices and has been extensively studied from Ancient times. Aristotle discusses metaphor in two of his major works: *Poetics* and *Rhetoric*. He mentions that metaphor 'makes learning pleasant'. In his book *Poetics*, Aristotle views metaphor as predominantly a substitution and transfer between two categories. Based on the type of transfer, he classifies metaphors into different groups (Fergusson, 2019). This theory is sometimes called the 'classical view' which became the dominant approach to the study of metaphor within the field of rhetoric.

Aristotle's views on metaphor continued to hold sway in philosophical discourse until they finally began to be questioned in the latter half of the twentieth century, after which interest in the study of metaphor resurfaced (Wood, 2015).

Example (2) is a metaphorical sentence with the replacement of one nominal for another:

(2) The suspect was described as a lone wolf.

Here, *lone wolf* is a metaphor referring to solitude, nonconformism and introversion. It is a substitute for words like *loner*, or *independent*. Substitution-based view of metaphors, however, have been deemed inadequate and lacking in rigour in recent literature and new theories have appeared to address the perceived shortcomings.

There are conflicting views on the prevalence and history of metaphor. For instance, Liberman (2019) suggests metaphor has only become common in European languages after the advent of the renaissance. Lakoff and Johnson (2008), however, have a unique viewpoint in that they view metaphor as a tool for conceptual representation and an indispensable part of human thought, thus elevating this phenomenon beyond its linguistic confines.

2.2.2 Irony

Irony, from the Greek root eirōneia ("simulated ignorance") (Oxford Learner's Dictionaries, 2020b), refers to an utterance whose literal meaning is radically

different from what is intended, sometimes to the point of contrast (Wilson and Sperber, 1992).

A closely connected term is **sarcasm**, which refers to instances where the speaker deliberately states the opposite of what is intended in order to emphasise his/her point and is usually associated with a negative remark, bringing an element of spite into the overall sentiment. It can also involve overstating or understating a fact.

For all practical purposes, sarcasm can be considered a subset of irony, and in NLP literature they are commonly lumped together. For this reason, unless otherwise stated, we refer to both phenomena using the term irony.

At the core of irony is a shared knowledge between the speaker and the audience, which is sometimes called "principle of inferability" (Kreuz and Link, 2002). The speaker assumes this common understanding when making an ironic statement, and within the message, there is some misplaced element to signal the existence of irony.

The advent of social media and in particular, micro-blogging services like Twitter, has opened unprecedented avenues for Internet users around the world to express themselves and share their opinions widely. The spatial constraints of micro-blogging, coupled with the non-standard language, heavy use of chat abbreviations, emojis, emoticons, memes, and other visual elements pose a challenge to automatic processing of text in such contexts. Users also tend to use playful and creative language on these platforms, which includes peppering comments with irony, especially when talking about sen-

sitive topics like politics.

In some cases, there are obvious markers signalling the presence of irony in social media text (e.g. capitalisation, punctuation, hyperbolic words, and certain morpho-syntactic patterns like interjections and tag questions). However, in many cases, ironic intent becomes apparent only after consideration of a broader context including the topic of the conversation, the particular character of the user, the history of the conversation up to the point of the comment, and common sense and general connotative knowledge (Ghosh et al., 2018; Van Hee et al., 2018b).

As social media text is rife with irony, most available sarcasm/irony datasets in NLP are related to social media platforms like Reddit and Twitter (Reyes et al., 2013; Wallace et al., 2014; Khodak et al., 2017). The experiments in Chapter 3, therefore, will be performed on a Twitter dataset.

2.2.3 Multiword Expressions (MWEs)

Sag et al. (2002) in their seminal work on Multiword Expressions (MWEs) define them as "idiosyncratic interpretations that cross word boundaries (or spaces)". MWEs can be decomposed into multiple simplex words and are "lexically, syntactically, semantically, pragmatically and/or statistically idiosyncratic" (Baldwin and Kim, 2010). Examples include compound nouns, idioms, verb-particle and light verb constructions, among others.

Some MWEs are fixed expressions, while others demonstrate a range of behaviour that can make their computational treatment challenging. These

include (but are not limited to) degrees of semantic opacity (i.e. semantics of whole cannot be inferred directly from the meaning of the components), syntactic, and lexical variability (i.e. same expressions can be used with slightly different words and syntactic structure), and discontinuity (i.e. MWEs can be broken into parts separated by gaps).

MWEs have different degrees of non-compositionality. This means, some MWEs can be decomposed to their individual parts, and there is a high correlation between the semantics of the constituents and the overall meaning (e.g. *traffic light*). These are also the ones that demonstrate a higher degree of syntactic variability. A large number of MWEs, on the other hand, are not decomposable to their constituents and they range from wholly fixed (e.g. *kick the bucket*) to partially variable (e.g. compare *let the cat out of the bag* with *the cat was let out of the bag*) (Baldwin et al., 2003; Sheinfux et al., 2017).

Researchers make a distinction between MWEs and regular phrases composed of multiple words. Even though the components of an MWE can often be separated, the distinction lies in the **semantic idiosyncrasy** of MWEs, which is another common term denoting the same notion of non-compositionality and is used to differentiate MWEs from other linguistic units (Fazly and Stevenson, 2007).

In this work, we specifically focus on verbal MWEs, which have a more transparent connection with verbal metaphors.

2.2.4 The Commonalities Among The Selected Cases

The byproduct of diversion from literal sense is that, in order to decode the intended meaning, consideration of context and a non-compositional semantic analysis is needed(Sikos et al., 2008). Therefore, the most important common characteristic among the different forms of non-literal language is the need to consider the textual context within which the utterance is made. This is in contrast to literal language where the meaning of the whole can be formed from the individual components by a compositional process regardless of the specific context the utterance appears in.

Expression	Metaphorical	Fixed MWE	Ironic
Time flies	✓	✓	✗
Elephant in the room	✓	✓	✗
Towering figure	✓	✗	✓
Break a leg	✓	✓	✓
Pigs might fly	✓	✓	✓

Table 2.1: Examples that show how MWEs, irony, and metaphor can overlap

Sometimes the lines separating these three phenomena become blurry (Table 2.1). Many MWEs are fixed expressions that can be considered metaphorical. Expressions like *time flies*, *winning hearts and minds*, *red herring*, *pig in a poke*, or *fall in love* are all examples of this category. Sometimes the original imagery is lost over time due to extensive use, and the speaker uses the metaphor as a set phrase without thinking about its earlier connotation. Such expressions are called **dead metaphors** (Pawelec, An-

drzej, 2020) which include all the above examples. There is a discussion in the literature as to whether this imagery is always dead (Lakoff, 1987) or whether such expressions can be considered metaphorical at all (Black et al., 1979). Regardless of the existence of the link to the original sense, it is clear there is an overlap between metaphor and idioms. Expressions like *elephant in the room*, *cut to the chase*, and *bull market* are clearly metaphorical and also fixed MWEs.

Same links exist between irony and the other two cases. Consider a situation in which a person looks at the terrible handwriting of a doctor (Figure 2.1) and says *What delicate lacework!* (Popa-Wyatt, 2017). This example can be considered a case of ironic metaphor. A similar example is when someone refers to an ineffective politician as a *towering figure* (Popa-Wyatt, 2017).

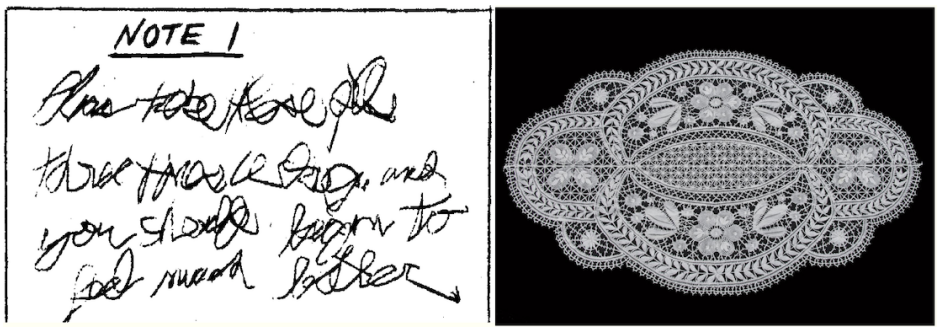

Figure 2.1: A messy piece of handwriting is metaphorically linked to lacework

An expression like *break a leg* is an ironic MWE, as it is referring to a theatrical superstition in which one wishes others good luck by saying something negative. *Pigs might fly* is an example of an ironic metaphorical

MWE, sitting at the intersection of the three cases of non-literal language.

2.2.5 The Differences Among The Studied Cases and Possible Issues

The most significant difference between MWEs and the other two cases is that irony and metaphor belong to creative language and have a higher degree of flexibility. MWEs are lexicalised and often used in writing and conversation in a predictable way, which makes them easy to detect by a human. Irony and metaphor can be sometimes harder to identify as they can appear in different forms and be very subtle and nuanced. Some could be understood by a limited number of people who share a piece of common knowledge with the speaker. Some metaphors might be culturally specific to an area and not understood by people who speak the same language but have a background rooted in a different culture, history, and mythology.

Understanding irony might depend on paralinguistic context. For example, consider the tweet *The best thing I learned in school: https://bit.ly/2VttXLD*, which includes a link to a photo of a coffee shop. Understanding ironic intent, in this case, requires some reasoning beyond the immediate linguistic context. All examples of situational irony are of this category. MWEs, on the other hand, are strictly linguistic and do not require reasoning of this kind.

Since metaphor is, in its essence, a comparison between two domains, it can happen outside the realm of language and in visual form. Figure 2.2

is an example where metaphor is understood by linking concepts through visualisation.

Figure 2.2: A metaphorical visualisation of anger (Pavlov, 2009)

In this thesis, the focus is on examples of metaphor and irony that are understood through linguistic context alone. The datasets used reflect this intent, and with very few exceptions, most studied examples are of this kind.

2.3 The Machine Learning Concepts Used in The Experiments

In the design of the experiments to model non-literal language, we made use of numerous machine learning tools and techniques. Before we delve into the details of the experiments in the upcoming chapters, we introduce the most important ones below, starting from the classic (non-neural) tools and later introduce the deep learning components.

2.3.1 Logistic Regression

Logistic Regression (LR)[1] is a standard classification technique[2] used to model probabilities of two or more classes of events. It was originally invented to study the patterns of population growth and analysis of chemical reactions(Cramer, 2002). This classifier depends on the logistic function (2.1) which is the basis for many other classification methods:

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2.1}$$

Let X be a random variable representing the input values and Y another random variable with values limited to the set $\{-1, 1\}$, which denotes two target classes representing qualitative values. Like any other classification method, the goal of LR is to output a probability which shows how likely Y belongs to a certain class given each value of X (i.e. $p(X) = Pr(Y = 1|X)$ or alternatively $Pr(Y = -1|X)$).

Using 2.1 and with coefficients β_0 and β_1, LR can fit on the training data with 2.2, which is mathematically equivalent with 2.3:

$$p(X) = \frac{e^{\beta_0+\beta_1 X}}{1 + e^{\beta_0+\beta_1 X}} \tag{2.2}$$

$$\frac{p(X)}{1 - p(X)} = e^{\beta_0+\beta_1 X} \tag{2.3}$$

[1]Description of algorithms at 2.3.1 and 2.3.2 are, for the most part, based on James et al. (2013). For the generalisation of LR beyond two classes, we use the notation by Smith (2011).

[2]The word 'regression' can be confusing here since the technique is used for classification.

The left hand side of 2.3 is called the *odds* which can range anywhere from zero to infinity. Having taken the *log* from both sides we arrive at what is called *logits* (or *log-odds*) which is linear with respect to X:

$$log(\frac{p(X)}{1-p(X)}) = \beta_0 + \beta_1 X \tag{2.4}$$

LR uses a method called *maximum likelihood* to estimate the coefficients β_0 and β_1 in 2.2 based on the given training data. The intuition behind this method is to estimate values such that the predicted probabilities at 2.2 correspond as closely as possible with the observed classes in training. Values close to zero correspond to the negative class and the ones close to 1 to the positive. To perform this estimation we use the likelihood function given at 2.5. The objective is to choose these two values (β_0 and β_1) so as to maximise ℓ:

$$\ell(\beta_0, \beta_1) = \prod_{i:y_i=1} p(x_i) \prod_{i':y_{i'}=0} (1 - p(x_{i'})) \tag{2.5}$$

This procedure is rarely done by hand and is usually automated using statistical and machine learning libraries. In this work, we make use of LR for the experiments in Chapter 3 and perform the above procedure using the Python library sklearn (Pedregosa et al., 2011).

In reality, there are often several different predictors affecting the outcome, which corresponds to a p-dimensional $X = (X_1, X_2, ..., X_p)$. We can simply rewrite 2.2 to account for that and the same maximum likelihood procedure can be used to estimate β_i $(i = 0, 1, 2, ..., p)$:

$$p(X) = \frac{e^{\beta_0+\beta_1 X_1+...+\beta_p X_p}}{1 + e^{\beta_0+\beta_1 X_1+...+\beta_p X_p}} \tag{2.6}$$

Equation 2.6 follows that 2.4 can also be rewritten as the following:

$$log(\frac{p(X)}{1-p(X)}) = \beta_0 + \beta_1 X_1 + ... + \beta_p X_p \tag{2.7}$$

LR can also be generalised beyond binary classification where there are K different target classes. This is sometimes called *multinomial* LR. In this scenario, Y ranges over the discrete finite set $\mathcal{Y} = \{y_1, y_2, ..., y_K\}$ and each $y \in \mathcal{Y}$ has its own vector of coefficients $\boldsymbol{w_y}$.

$$p(Y = y|X = x) = \frac{e^{\boldsymbol{w_y}^T x}}{\sum_{k=1}^{K} e^{\boldsymbol{w_k}^T x}} \tag{2.8}$$

2.3.2 Support Vector Machines

Support Vector Machine (SVM) is a powerful classification method which is based on the *Support Vector Classifier (SVC)*, which in turn is a generalisation of the *Maximal Margin Classifier (MMC)*. In order to introduce the terms needed to explain SVM and understand the justification behind its mechanics, we will first describe MMC without going into the mathematical details of its optimisation objective.

MMC is predicated on the assumption that the two target classes[3] can be linearly separated. This results in a mathematically elegant albeit practically

[3]As in the case of LR, we start with the binary classification scenario and leave the multi-label extension to the end.

constrained method which can be visually described in terms of *hyperplanes*.

In a p-dimensional space, a hyperplane is defined by 2.9, where β_0, β_1,..., β_p are the coefficients. If a point $X = (X_1, X_2, ..., X_p)^T$ satisfies this equation, it is said to be located on the hyperplane.

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_p X_p = 0 \tag{2.9}$$

It is easy to visualise a hyperplane in 2-D and 3-D scenarios (a line and a plane respectively). In general terms, a hyperplane in a p-dimensional case is a "flat affine subspace of dimension $p - 1$" with *affine* implying that the hyperplane need not pass from the point of origin (James et al., 2013).

In geometrical terms, a hyperplane can divide the p-dimensional space into two halves. If a point is not precisely on the hyperplane, it is therefore situated on one of its two possible sides. This is determined by the sign of the left-hand side of 2.9 when we plug in X and get a non-zero number. This is called a *separating hyperplane* which is the basis of MMC.

The minimum perpendicular distance from the hyperplane to the observation points is called a *margin*. If a separating hyperplane exists, it can be shown that there are infinitely many possible hyperplanes that achieve the same effect. MMC's objective is to find the best possible separating hyperplane, that is the one with the largest margin.

Samples that are located on the margin are called *support vectors*. A small movement in the location of support vectors can immediately change the position of the separating hyperplane. It can be shown that support

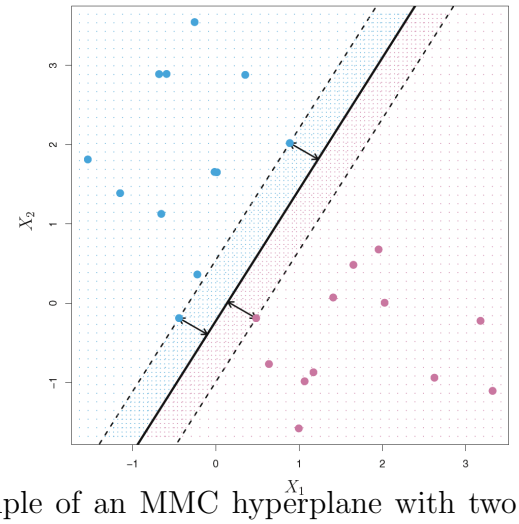

Figure 2.3: Example of an MMC hyperplane with two classes from James et al. (2013). Points on the dashed lines are support vectors.

vectors are the only observations that affect the classifier, and changes in the remainder of the training points are unimportant as long as they occur outside of the margin. An example of an MMC hyperplane in 2-D can be seen in figure 2.3.

There are two major shortcomings with MMC. On the one hand, it is too sensitive to small changes in the support vectors making it less robust to changes in datasets and prone to overfitting. On the other hand, its assumption that a perfect separating hyperplane exists is rarely the case in practice rendering the classifier virtually useless for many real-world problems. The MMC's insistence on accommodating all data points in either side of its hyperplane can result in margins becoming too thin, which undermines the reliability of the model.

SVC addresses these issues by changing the underlying assumption in MMC. Instead of attempting to classify all the data points perfectly, SVC's objective is to find a hyperplane that classifies most of the observations while allowing for some errors and exceptions. An imperfect hyperplane would be more robust to individual changes and less likely to overfit on the training data.

SVC achieves this objective by making some changes to equation 2.9. Let M represent the margin and there are n labels $y_1, y_2, ..., y_n \in \{-1, 1\}$. The objective is to maximise M while 2.10 holds for any i:

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_p x_{ip}) \geq M(1 - \epsilon_i) \tag{2.10}$$

The ϵ_is are called *slack variables* which are non-negative values that allow for a certain amount of deviation and error in the classification. C in 2.11 is the total error allowance:

$$\sum_{i=1}^{n} \epsilon_i \leq C \tag{2.11}$$

In simple terms, C controls for the total amount of 'sloppiness' we can allow in the classification. It is sometimes called the *budget*. A zero C would revert the classifier back to MMC as no misclassification would be allowed. A large C would reduce the predictability of the model and make it too biased. This variable is usually set as a hyperparameter using a development set.

Once the optimum M is achieved in training time, for any given x^* in the test set, we can classify the observation based on the sign of $f(x^*) = \beta_0 + \beta_1 x_1^* + \beta_2 x_2^* + ... + \beta_p x_p^*$.

Similar to MMC, here the classifier is also solely reliant on the support vectors, which in this context are the points that are on the margin plus the ones that violate it. The more lenient margin in SVC is called a *soft margin*.

Despite the improvements over MMC, SVC still suffers from one major limitation: It assumes that the boundary separating the classes is linear. SVM addresses this limitation by augmenting SVC to allow for non-linear boundaries. SVM achieves this by increasing the feature space through the use of *kernels*. A kernel is a function that measures the similarity between each two observations.

It can be shown that the SVC classifier can be represented as the inner product of observations (equation 2.12), where $\langle x, x_i \rangle = \sum_{i=1}^{r} a_i b_i$ is the inner product of two r-vectors and the n different α_i are parameters of the model.

$$f(x) = \beta_0 + \sum_{i=1}^{n} \alpha_i \langle x, x_i \rangle \tag{2.12}$$

In SVM, the inner product is replaced with a kernel function $K(x_i, x_i^{'})$ that results in non-linear decision boundaries. Polynomial and radial kernels are two common examples of non-linear kernels used in SVMs.

SVM and LR tend to produce similar results on most tasks. However, SVMs might perform slightly better in cases when the classes are clearly separated. Because of these small differences, they have been used in ensemble scenarios in Chapter 3. Just like LR, SVM is not naturally designed for multi-label classification scenarios. In such cases, two possible extensions to SVM exist, namely, *one-versus-one* and *one-versus-all* approaches.

In a one-versus-one or *all-pairs* approach with K class labels, $\binom{K}{2}$ different binary SVMs are constructed for each pair of labels. At test time, each observation is passed through all these classifiers and is assigned the label that it most often receives.

In the one-versus-all scenario, K different binary SVMs are constructed, in which each class is compared against $K-1$ other classes. The test observation is assigned the label that results in the highest score from these K classifiers.

2.3.3 Recursive Feature Elimination

Recursive Feature Elimination (RFE) is a robust feature ranking technique that helps a model select the most relevant subset of the feature space. It was first introduced in 2002 by Guyon et al. (2002) to select genes in a cancer classification task using SVM. It has since become a staple feature selection method and used with many different classification and regression models (Kuhn and Johnson, 2019).

Reducing the dimensionality of the feature space is a critical step to ensure the model is not overfitting on the training set, and it subsequently improves generalisability. This is especially important in cases when the feature space is large compared to the number of training samples. A pruning method based on exhaustive enumeration of all possible subsets of features would only be feasible when the feature set is not very big (i.e. tens rather hundreds or thousands of features). However, in many practical problems, this is not

the case. For example, in the original RFE paper, there are thousands of gene features, and an exhaustive method would lead to what the authors call a 'combinatorial explosion'(Guyon et al., 2002). For this reason, when dealing with an extensive feature set, feature selection methods are usually based on *greedy algorithms*(Kotsiantis, 2011)[4].

In RFE, the model is initially trained with all the features, and subsequently, the features are ranked and the least effective one is pruned. This process is recursively done until the desired number of features is reached. The ranking criterion can be different in one algorithm to another. In figure 2.4, we see the pseudo-code of the algorithm in the case of SVM, where the coefficients of the separating hyperplane are the basis for feature ranking.

Data : Dataset with p^* variables and binary outcome.
Output: Ranked list of variables according to their relevance.

Find the optimal values for the tuning parameters of the SVM model;
Train the SVM model;
$p \leftarrow p^*$;
while $p \geq 2$ **do**
  $SVM_p \leftarrow$ SVM with the optimized tuning parameters for the p variables and observations in **Data**;
  $w_p \leftarrow$ calculate weight vector of the SVM_p $(w_{p1}, \ldots, w_{pp})$;
  $rank.criteria \leftarrow (w_{p1}^2, \ldots, w_{pp}^2)$;
  $min.rank.criteria \leftarrow$ variable with lowest value in $rank.criteria$ vector;
  Remove $min.rank.criteria$ from **Data**;
  $Rank_p \leftarrow min.rank.criteria$;
  p ← p - 1 ;
end
$Rank_1 \leftarrow$ variable in **Data** $\notin (Rank_2, \ldots, Rank_{p^*})$;
return $(Rank_1, \ldots, Rank_{p^*})$

Figure 2.4: SVM-RFE algorithm using the linear kernel in a model for binary classification (Sanz et al., 2018).

[4]A greedy algorithm is one that makes the locally optimum choice at each step to approximate the global optimum.

2.3.4 Conditional Random Fields

Conditional Random Fields (CRFs) are a special case of *log-linear* models which can be considered an extension of LRs (2.3.1). To understand CRFs, it is important to learn about log-linear models and the way they work. Elkan (2008) and Sutton et al. (2012) are two resources where one can learn about CRFs, and this introduction is mostly based on the former. CRFs are the basis for experiments conducted in Chapter 4.

A log-linear model assumes that for any $x \in X$ the probability of label $y \in Y$ can be calculated from equation 2.13[5].

$$p(y|x; w) = \frac{exp \sum_{j=1}^{J} w_j F_j(x, y)}{Z(x, w)} \tag{2.13}$$

$F_j(x, y)$ and $Z(x, w)$ are respectively called *feature* and *partition* functions. The feature function, mathematically a mapping $F_j : X \times Y \rightarrow \mathbb{R}$[6], can be intuitively understood as a measure of compatibility between x and y. As there are J different feature functions, each represents a separate measure of compatibility. Feature functions are usually defined using templates based on a combination of binary presence/absence indicators. Each w_j is a corresponding learnable weight determining the importance of the feature functions. Feature functions are defined beforehand by a human, but weights are automatically learned during training.

The partition function Z is a constant whose value is calculated based on

[5] e^x can be alternatively written as $exp(x)$

[6] An important special case is boolean, where the formulation is $F_j : X \times Y \rightarrow \{0, 1\}$

x and y:

$$Z(x, w) = \sum_{y' \in Y} exp \sum_{j=1}^{J} w_j F_j(x, y') \tag{2.14}$$

To determine what $\hat{y}$ is best given each x in 2.13, we can ignore the constant in the bottom and find one that maximises the numerator[7] (Eq. 2.15).

$$\hat{y} = argmax_y p(y|x; w) = argmax_y \sum_{j=1}^{J} w_j F_j(x, y) \tag{2.15}$$

In Eq. 2.13, the linear combination can produce any real-valued number. The exponential operator converts this number to a positive value, and the division by the constant ensures that value is a valid probability.

In Section 2.3.1 we looked at the logistic sigmoid function (Eq. 2.1) which normalised values to a [0,1] range in the task of binary classification. Here, we have several possible labels $\hat{y}$, and this normalisation is done through the *softmax function* which in general can be formulated as

$$softmax(x) = \frac{\exp(x)}{\sum_{x'} \exp(x')} \tag{2.16}$$

Looking back at equations 2.13 and 2.14, and comparing them to 2.16 we can see how the softmax function here is applied.

With this preamble about log-linear models, we can now focus on linear-chain CRFs. In NLP, CRFs can be used for sequence labelling where each token receives a label. CRFs are particularly helpful in *structured prediction*

[7]The exponential operator can also be safely ignored as it plays no role in determining what $\hat{y}$ maximises the numerator

tasks where there are dependencies among the tags comprising each label. Similar to log-linear models, here sentences are represented through a host of different feature functions that reflect semantic and syntactic information about each word in the sentence.

If we consider each sentence as $\bar{x}$, its corresponding label is denoted as $\bar{y}$, which in turn is comprised of i different tags y_i. These tags are necessarily taken from a finite set. In order to adapt the log-linear model to the constraints of structured prediction, the feature functions from Eq. 2.13 need to be slightly tweaked. In the new formulation, F_j will depend on the entirety of the sequence as it is defined by summation over $\bar{y}$:

$$F_j(\bar{x}, \bar{y}) = \sum_{i=1}^{n+1} f_j(y_{i-1}, y_i, \bar{x}, i) \tag{2.17}$$

Training the CRF in this case would correspond to finding the best w that would provide us with the best prediction:

$$\hat{y} = argmax_{\bar{y}} p(\bar{y}|\bar{x}; w) \tag{2.18}$$

This optimisation problem is usually solved using log conditional likelihood (LCL). For details of the inference algorithm, please consult Elkan (2008).

2.3.5 Convolutional Neural Networks

Convolutional Neural Networks(CNNs) are a class of neural network architectures that are comprised of convolving filters that generate features at

the local level. They are not standalone and are used as part of a larger network. CNNs were initially introduced within the field of computer vision (LeCun et al., 1995); however, they have proved to be useful across a wide range of tasks in NLP as well, starting by the pioneering work of Collobert et al. (2011). We can think of CNNs as n-gram feature generators (Kim, 2014). They are usually applied before a recurrent or feedforward layer and jointly trained with the rest of the layers during training. In this thesis, we make heavy use of CNNs in Chapter 5 and integrate them within larger architectures to act as feature generating front-ends.

In tasks like MWE identification, irony detection, or metaphor classification, for the model to make the right decisions, it needs to take local ordering into account. For this reason, a model like bag-of-words which completely disregards ordering would not result in effective representation. The power of CNNs is in their ability to preserve these nuanced differences at the local level, which is the incentive behind using them as potent feature generators in CNN-RNN and other varieties of hybrid models (Goldberg, 2017).

In the original formulation, CNNs are sequential by default and look at contiguous spans of text. However, CNN blocks can be defined on top of one another to form a complex network, allowing for the model to also consider non-contiguous n-grams. It is also possible to design CNNs that can periodically ignore certain positions when scanning over sequences of text. It is important to note that in the realm of text, we think of convolution as a 1D operation, but in vision tasks, the model convolves over a 2D grid.

CNN is usually accompanied by a pooling operation; however, in sequence labelling, this is not appropriate as it would reduce the resolution of the representation and lead to loss of valuable information (Strubell et al., 2017). For this reason, this operation is not applied in any of the CNN modules in our experiments. We will, therefore, not discuss pooling here and in the remainder of this section will focus on 1D convolution over a sequence of text. The mathematical formulation we use here is based on Goldberg (2017).

If we have a sequence of n words $w_{1:n} = w_1, w_2, ..., w_n$, let the corresponding representation for each word be denoted by $\mathbf{E}_{[w_i]} = \mathbf{w_i}$ which is a d_{emb}-dimensional word embedding. A 1D convolution is a sliding window of a variable size k, which scans over the entire sequence and applies what is known as a *filter* to each window. Applying the filter means performing dot product with the weight vector $\boldsymbol{u}$, followed by a non-linear activation. Let $\oplus(\boldsymbol{w}_{i:i+k-1})$ be the concatenation of word vectors $\boldsymbol{w}_i, \boldsymbol{w}_{i+1}, ..., \boldsymbol{w}_{i+k-1}$. For the kernel size k, we can show the concatenated vectors in the ith window with $\boldsymbol{x}_i$ where

$$\boldsymbol{x}_i = \oplus(\boldsymbol{w}_{i:i+k-1}) = [\boldsymbol{w}_i; \boldsymbol{w}_{i+1}; ...; \boldsymbol{w}_{i+k-1}], \boldsymbol{x}_i \in \mathbb{R}^{k.d_{emb}} \tag{2.19}$$

For each application of the filter to the ith window, the resulting value is a scalar p_i:

$$p_i = f(\boldsymbol{x}_i \odot \boldsymbol{u}) \tag{2.20}$$

$$\boldsymbol{x}_i = \oplus(\boldsymbol{w}_{i:i+k-1}) \tag{2.21}$$

where $p_i \in \mathbb{R}, \boldsymbol{x}_i \in \mathbb{R}^{k.d_{emb}}, \boldsymbol{u} \in \mathbb{R}^{k.d_{emb}}$ and f is a non-linear activation function.

In practice, l different filters $\boldsymbol{u}_1, \boldsymbol{u}_2, ..., \boldsymbol{u}_l$ are used to represent each window. They can be put together in the form of a matrix $\boldsymbol{U}$ and an added bias $\boldsymbol{b}$. Therefore Eq. 2.21 can be rewritten as

$$p_i = f(\boldsymbol{x}_i \odot \boldsymbol{U} + b) \tag{2.22}$$

where $p_i \in \mathbb{R}^l \quad \boldsymbol{x}_i \in \mathbb{R}^{k.d_{emb}} \quad \boldsymbol{U} \in \mathbb{R}^{k.d_{emb} \times l} \quad \boldsymbol{b} \in \mathbb{R}^l$. Each p_i, is a vector of l values that corresponds with the ith window.

2.3.6 Graph Convolutional Neural Networks

The CNN architecture explored in Section 2.3.5 has become a staple component of many popular deep learning architectures. However, CNNs suffer from one major disadvantage, and that is the sequential nature of their kernels. In each pass over a k-word window, the convolution operation is applied to all the consecutive tokens. In cases where there are interlinked tokens separated by gaps, the default sequential CNN might not be able to capture vital dependencies which leads to poorer representation and overall performance.

There are different ways to mitigate this issue, two of which we will briefly discuss before moving on to Graph Convolutional Networks (GCNs). In standard CNN, the kernel is moved one step at a time, with indices 1,2, ... In this case, we say that the *stride* is 1. However, the kernel need not be covering every single token in the sentence and can stride with higher values. For instance, if this value is k ($k > 0$), windows will start at indices 1, $1 + k$,

$1 + 2k$, etc. The output of CNN would be shorter in length compared with the original sentence.

Another way to capture non-sequential dependencies in CNNs is to use *dilated convolutions* (Strubell et al., 2017). In this architecture, CNNs are stacked on top of each other with a *dilation* value of greater than 1 ($\delta > 1$), which means the kernel skips over δ tokens at each time step. Let c_t be the output of CNN and x_t be the token at step t. Then the output of a CNN with dilation δ can be computed by

$$c_t = W_t \bigoplus_{k=0}^{r} x_{t \pm k\delta} \tag{2.23}$$

These remedies, while effective in some tasks, suffer from this fundamental problem that the ignored tokens are chosen simply because of their position in the sentence, and there is no way to know beforehand what informative token to keep and what to ignore. That is why GCNs are a potentially superior way to handle such cases, especially when we can build up a graph beforehand, where nodes would identify the informative links between each two tokens and show where tokens might not have any inter-dependencies.

GCN is defined as a directed multi-node graph $G(V, E)$ where $v_i \in V$ and $(v_i, r, v_j) \in E$ are entities (words) and edges (relations) respectively. By defining a vector x_v as the feature representation for the word v, the convolution equation in GCN can be defined as a non-linear activation function f

and a filter W with a bias term b as:

$$c = f(\sum_{i \in r(v)} W x_i + b) \tag{2.24}$$

where $r(v)$ shows all words in relation with the given word v in a sentence, and c represents the output of the convolution.

In the experiments conducted in this thesis at Chapter 5, we focus on the application of GCN to sequence labelling in tasks where syntactic information is encoded in the form of adjacency graphs. GCN, then, performs convolution based on these relations.

Following Kipf and Welling (2017) and Schlichtkrull et al. (2017), we represent graph relations using adjacency matrices which act as mask filters for the inputs. Words are linked using the dependency parse tree derived from the target sentence. Since we do sentence by sentence sequence labelling, there is an adjacency matrix representing relations among words (as nodes of the dependency graph) for each given sentence. We define the sentence-level convolution operation with filter W_s and bias b_s as follows:

$$C_s = f(W_s X^T A + b_s) \tag{2.25}$$

where X, A, and C are the representation of words, adjacency matrix, and the convolution output, all at the level of sentence. The above formalism considers only one relation type; however, depending on the application, multiple relations can be defined.

Kipf and Welling (2017) constructed separate adjacency matrices corresponding to each relation type and direction. Given the variety of dependency

relations in a parse tree (e.g. obj, nsubj, advcl, conj, etc.), and per-sentence adjacency matrices, we would end up with an over-parametrised model in a sequence labelling task. In this work, we simply treat all relations equally, but consider only three types of relations: 1) the head to the dependents, 2) the dependents to the head, and 3) each word to itself (self-loops). The final output is obtained by aggregating the outputs from the three relations.

In Chapter 6, we address the lossy nature of this conversion by introducing attention-based GCNs where self-attention (2.3.8) is used to induce further relations among components of a sentence.

2.3.7 Long Short Term Memory Networks

Long Short Term Memory networks (LSTMs) are a class of Recurrent Neural Networks (RNNs) developed in 1997 by Hochreiter and Schmidhuber (1997). Standard RNNs are in practice limited in the scope of context they can cover, since the influence of a given input either diminishes or overly scales up as it goes through the recurrent connections in the network. This is known as the *Vanishing Gradient Problem* (Hochreiter, 1998). An RNN is, therefore, likely to "forget" a word that is located at the beginning of a long sentence and this limitation can adversely affect performance. LSTM is an attempt, among many in history, to address this problem and has proved relatively superior compared to its predecessors. For the description of LSTM in this section, I primarily rely on Graves (2012) and Sutskever (2013).

LSTM is comprised of recurrently connected components known as *mem-*

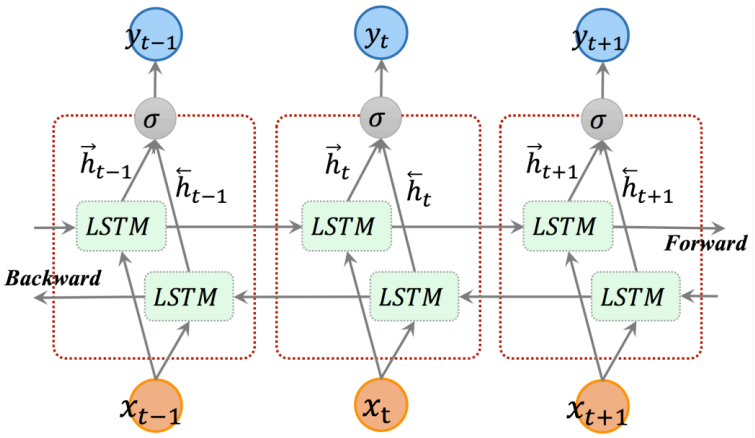

Figure 2.5: Unfolded architecture of bidirectional LSTM(Cui et al., 2018). 'σ' is the function that consolidates the outputs from each of the two LSTMs

ory units. Each unit has one or more connected memory cells and three auxiliary gating units, namely, the input, output, and forget gates. The idea behind these gates is to retain gradient information over a more extended period of time. For example, if the input gate is closed, new information can not come through and override what is recorded up to that point. The preserved information can then be passed on towards the end of the sequence by opening the output gate.

Figure 2.6 is a schematic demonstration of how LSTM gates can preserve information present at the beginning of the sequence all the way to the end. In this example, every memory unit is assumed to consist of only one memory cell. The intensity of shading in each cell is related to sensitivity to the information from the input at timestep 1. Therefore, white nodes are completely insensitive to the input, and black nodes are fully sensitive. As long as the forget gate is open and the input gate is closed, the information is preserved

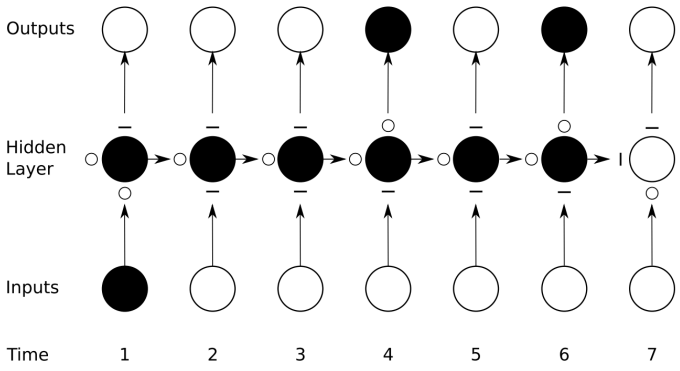

Figure 2.6: Preservation of gradient information by LSTM (Graves, 2012). For simplicity, it is assumed that every gate can only have two states, namely open ('∘') and closed ('–'). In reality, the activation is a real number between 0 and 1.

in the network, and LSTM is said to 'remember' the first input.

variable name	description
i_t^g	$[0,1]^N$-valued vector of input gates
i_t	$[-1,1]^N$-valued vector of inputs to the memory units
o_t	$[0,1]^N$-valued vector of output gates
f_t	$[0,1]^N$-valued vector of forget gates
v_t	$\mathbb{R}^v$-valued input vector
h_t	$[-1,1]^h$-valued conventional hidden state
$\tilde{m}_t$	$\mathbb{R}^N$-valued memory state available to the rest of the LSTM
m_t	$\mathbb{R}^N$-valued state of the memory units
z_t	the output vector

Table 2.2: Description of the variables used by LSTM(Sutskever, 2013). m_t is the input gate that allows for the update of the memory unit and $\tilde{m}$ is the output gate which controls what information can leave the unit.

With this demonstration, we can now formally define LSTM. Let LSTM have N memory units. In each timestep t, LSTM generates a set of vectors whose values are determined by the following equations (for the description

of the variables used, refer to Table 2.2):

$$h_t = tanh(W_{hh}h_{t-1} + W_{hv}v_t + W_{hm}\tilde{m}_{t-1}) \tag{2.26}$$

$$i_t^g = sigmoid(W_{igh}h_t + W_{igv}v_t + W_{igm}\tilde{m}_{t-1}) \tag{2.27}$$

$$i_t = tanh(W_{ih}h_t + W_{iv}v_t + W_{im}\tilde{m}_{t-1}) \tag{2.28}$$

$$o_t = sigmoid(W_{oh}h_t + W_{ov}v_t + W_{om}\tilde{m}_{t-1}) \tag{2.29}$$

$$f_t = tanh(b_f + W_{fh}h_t + W_{fv}v_t + W_{fm}\tilde{m}_{t-1}) \tag{2.30}$$

$$m_t = m_{t-1} \odot f_t + i_t \odot i_t^g \tag{2.31}$$

$$\tilde{m}_t = m_t \odot o_t \tag{2.32}$$

$$z_t = g(W_{yh}h_t + W_{ym}\tilde{m}_t) \tag{2.33}$$

In the Bidirectional LSTM (Bi-LSTM), there is an additional network that processes the sequence in reverse, and the output is generated by the concatenation[8] of the hidden states of the forward and backward LSTMs, as shown in Figure 2.5. Bi-LSTMs are baked into the core of the architecture presented at Chapter 5.

2.3.8 Multi-head Self-attention

Attention is a mechanism that first appeared in the context of sequence modelling to extend encoder-decoder (Cho et al., 2014; Sutskever et al., 2014)

[8]Summation, multiplication, and averaging are theoretically possible as well.

models in neural machine translation where the task is to map pairs of sequences that do not necessarily have the same length. Attention has since found applications in many NLP tasks including sequence labelling.

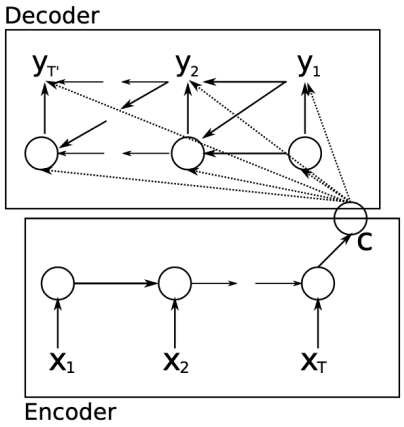

Figure 2.7: An illustration of RNN-based encoder–decoder (Cho et al., 2014)

In order to understand the incentive behind the attention mechanism, it helps to have a high-level understanding of the original RNN-based encoder-decoder model. As shown in Figure 2.7, the architecture is composed of two RNNs called 'encoder' and 'decoder' which are jointly trained. At each time step t, the encoder reads one symbol from the variable-size $X = (x_1, x_2, ..., x_T)$ and updates its hidden state $h_t = f(h_{t-1}, x_t)$, where f is a non-linear activation. When the scan is complete, the encoder module converts the entire sequence to a fixed-length vector representation c which is the same as the last updated hidden state. At each timestep t, the decoder computes its hidden state $h_t = f(h_{t-1}, y_{t-1}, c)$, based on the entire prior context (c), the previously predicted label (y_{t-1}), and the last hidden state (h_{t-1}). Based on h_t, the decoder predicts the label y_t conditioned on all previously

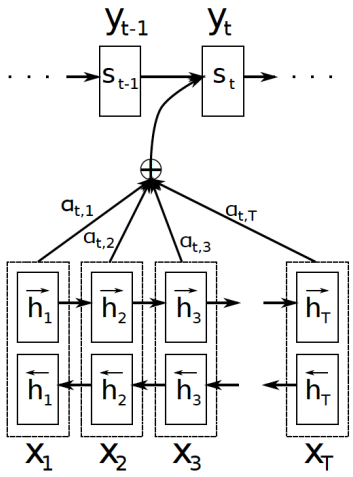

Figure 2.8: Overview of the attention-based encoder-decoder (Bahdanau et al., 2014). At timestep t, the decoder is about to generate target word y_ts

predicted labels and the context:

$$P(y_t|y_{t-1}, y_{t-2}, ..., y_1, c) = g(h_t, y_{t-1}, c) \tag{2.34}$$

where g is another activation that generates probabilities (e.g. softmax).

From Section 2.3.7, we know that RNNs are prone to losing track of vital information, especially when the sequence length increases. In other words, the summary of the sequence stored at the end of the encoder is likely to be biased towards the last few tokens, and since it is used to inform every generated label in the decoder, a poor representation can severely affect the performance of the model (Cho et al., 2014).

Having identified this bottleneck, Bahdanau et al. (2014) designed an upgraded version of the encoder-decoder model, in which for each label y_i that the decoder generates, it relies on a different c_i. Therefore, instead of

cramming everything into a single context vector, there is a different context dynamically generated at each timestep. The decoder defines the conditional probability $p(y_i|y_1, ..., y_{i-1}, X) = g(y_{i-1}, s_i, c_i)$ where s_i is an RNN hidden state at timestep i, computed by $s_i = f(s_{i-1}, y_{i-1}, c_i)$.

The encoder is a bi-directional LSTM (2.3.7) that maps the source sentence to a set of hidden states $(h_1, h_2, ..., h_{T_x})$. These are referred to as 'annotations'. At each timestep, the decoder computes $c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j$ as a weighted sum of the annotations using learnable parameters (Figure 2.8). In this way, it was claimed the decoder can adaptively select a subset of the annotations and in each timestep focus on or 'attend to' the most relevant part of the source sentence.

The attention mechanism has since become ubiquitous in NLP systems with many works integrating its different variants inside state-of-the-art models. A specific version of attention, namely, *self-attention* was popularised in 2017 by Vaswani et al. (2017). In this seminal work, a novel encoder-decoder architecture known as the 'Transformer' is introduced, which completely shelves RNNs and CNNs in favour of attention mechanisms. Unlike previous architectures where the decoder is the only component powered by attention, in the Transformer self-attention is repeatedly used in both encoder and decoder parts as the main building block. The overall architecture of the Transformer is not relevant to this thesis; however, we will discuss self-attention in detail as it has been used as a sub-module in some of the experiments in chapters 5 and 6.

The idea of self-attention was originally introduced by Cheng et al. (2016) in the task of machine reading[9] and was referred to as 'intra-attention'. Inspired by the incremental nature of language comprehension in humans, this work was an attempt to simulate a general-purpose machine 'reader'. The attention module was defined alongside an RNN to induce relations between elements of the same sequence.

The paper argued that in vanilla LSTM, when calculating the current hidden state h_t (Eq 2.26), LSTM only relies on the previous state h_{t-1} and not on any of the states before $t-1$, predicated on the assumption that the current state alone summarises everything seen up to that point. This conditional independence was cited as potentially problematic in longer sequences or when memory size is limited. Another shortcoming of LSTM, according to this work, is the lack of a mechanism to explicitly reason over structure and identify interdependencies among tokens in a sequence. The proposed model which was named LSTMN aimed to address these limitations by incorporating a memory and attention *within* a sequence encoder (hence the name intra-attention) with the idea to discover lexical relations between tokens in each sequence.

Vaswani et al. (2017) defined attention as a mathematical operation that maps a query and a set of key-value pairs to an output, where query, key, and value are all vectors. The output is a weighted sum of the values, and

[9]'Machine reading' was used in that paper as a broad term that included different tasks like language modelling, sentiment analysis, question answering, and natural language inference.

the weights are determined by a compatibility function that compares corresponding keys and queries.

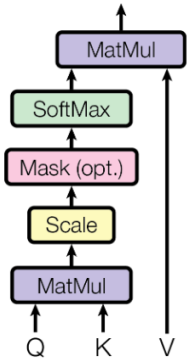

Figure 2.9: Scaled dot product attention (Vaswani et al., 2017)

Queries and keys are vectors of size d_k and the output is a vector of size d_v. The compatibility function is the dot product which corresponds with semantic similarity. All this mechanism does, is apply dot product on keys and queries, scale the output by $\sqrt{d_k}$, and pass it through a softmax layer (Fig. 2.9). Let Q, K, and V be a batch of queries, keys, and values. Then we can compute the output by

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{2.35}$$

Running multiple parallel attention mechanisms on separate subspaces of the same input can result in a richer representation. In this *multi-head* scenario, there are h different learned projections to perform several attention mechanisms in parallel. The outputs are finally concatenated together and the result is multiplied by a trained weight matrix W^O to produce the final

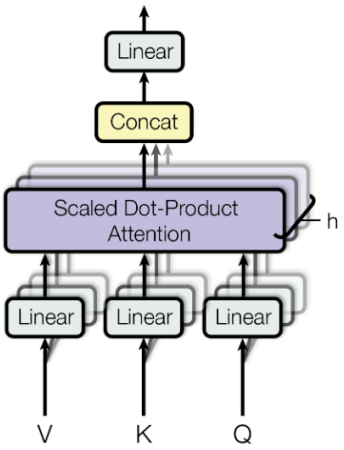

Figure 2.10: Multi-head self-attention (Vaswani et al., 2017)

output (Fig. 2.10). In mathematical terms:

$$MultiHead(Q, K, V) = Concat(head_1, head_2, ..., head_h)W^O \tag{2.36}$$

where $head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$. If the output for a single attention is d_{model}-dimensional, dimensions for the parameter matrices are the following: $W_i^Q \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^K \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{model} \times d_v}$, and $W^O \in \mathbb{R}^{hd_v \times d_{model}}$.

This type of attention has proved beneficial in many other tasks in NLP. However, since it operates independently from the rest of the architecture and loses positional information in the process of computing its output, it requires to receive positional encoding separately; otherwise, it would lower performance in tasks where global order matters. In our experiments, we used attention alongside CNN and LSTM modules to capture long-range dependencies.

2.3.9 Contextualised Embeddings

Neural embedding methods are used to represent words and sentences while capturing their syntactic and semantic nuances. These derived representations are then fed as input to other layers in a computational model. The quality of meaning representation is crucial in model performance, which is the reason why representation learning has attracted a considerable amount of interest in recent years. Pilehvar and Camacho-Collados (2020) overview most relevant concepts on embedding techniques.

In type-based embeddings like `word2vec` (Mikolov et al., 2013) and GLoVe (Pennington et al., 2014), a single vector is constructed for each token regardless of the different shades of meaning that it can acquire in various contexts. As an example, the token *can* is assigned a single fixed representation in both *He carried a metal can* and *He can swim very well*. This conflated representation can be potentially problematic in cases similar to the above example, where the different meanings of a single token can be entirely different. Contextualised embedding is an attempt to address the issue of polysemy and context-dependence by introducing dynamic token-based (rather than static type-based) methods to derive meaning representations. In addition to the classic type-based embeddings (Chapter 3), in this thesis we will make use of two prominent contextualised embedding methods, namely, ELMo (Peters et al., 2018a) and BERT (Devlin et al., 2018) (in chapters 5 and 6). For an explanation of the mathematical theories behind `word2vec` please consult

Goldberg and Levy (2014). In what follows we will have a brief look at the machinery of each of the contextualised models, starting with ELMo which can be described as the model that first popularised this technique [10]. Both of these models depend heavily on language modelling objectives. However, as we will see, ELMo uses an RNN-based encoder, but BERT relies on the Transformer model.

ELMo, which stands for 'Embeddings from Language Models' is a representation learning model that generates context-sensitive representation for a token, based on the internal states of a deep bidirectional language model called BiLM which is pre-trained on a large corpus. It employs a 2-layer Bi-LSTM encoder with residual connections to create a context-dependent representation for each word.

Figure 2.11 shows how the ELMo representation is constructed by the concatenation of three separate vectors: h_{k_1}, h_{k_2} which are the hidden states of the forward and backward LSTMs and x_k which is a static character-based representation for each token. The justification for the presence of two Bi-LSTM layers is that the higher level Bi-LSTM captures semantic information and the lower one deals with syntactic information. There are alternative ways to construct the final representation. It is possible to use only the top layer, for example, or simply average all the vectors or concatenate them all together.

[10] In the literature, there are influential earlier works like Melamud et al. (2016) that derive context-sensitive word representations based on Bi-LSTM language models. However, ELMo is the model that initially brought the most attention to this technique.

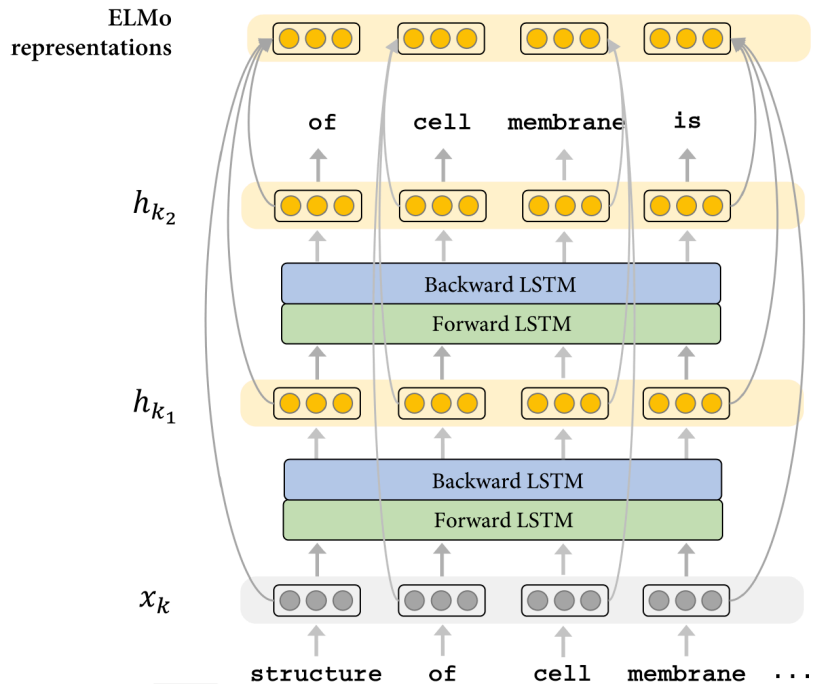

Figure 2.11: A high-level representation of ELMo from Pilehvar and Camacho-Collados (2020)

BERT, which stands for 'Bidirectional Encoder Representations from Transformers', is a model based on the Transformer encoder-decoder architecture. Whereas in ELMo, the language modelling task involved prediction of the next token, BERT introduced an alternative language modelling objective called Masked Language Modelling (MLM) which is inspired by the cloze test. MLM first randomly masks (i.e. replaces with a unique token) a certain percentage of words in the sentence and then the model is expected to guess the masked tokens.

Beside MLM, BERT has another objective called Next Sentence Prediction (NSP). In NSP, given a sentence A, the model is expected to guess whether B is the next sentence. To build a training dataset, the actual next

sentence is replaced with a random sentence half the time. This is a binary classification task whose goal is to inform the model of relationships between sentences which can prove useful in certain tasks like Question Answering.

For segmentation, BERT relies on the wordpiece algorithm (Wu et al., 2016). Using this method, words are broken into subword units and embeddings are learned for these smaller components. This reduces the vocabulary size, alleviates the issue of out-of-vocabulary words, and informs the model of morphological structures which can potentially help the model learn from cognates.

There are two ways to make use of pre-trained language models like ELMo and BERT. It is possible to derive a set of representations for a given data to be fed as embeddings into a separate task-specific architecture. This is referred to as the **feature-based** method. Alternatively, we can use the original pre-trained architecture of BERT and ELMo and build on top of that architecture. This is called the **fine-tuning** approach since it involves fine-tuning pre-trained parameters.

2.4 Summary

This chapter introduced the theoretical and technical foundation upon which the ensuing chapters are based. We looked at the definition of the linguistic terms that will be the focus in the thesis and overviewed the machine learning models and methods that are employed in our experiments. We defined metaphor, MWEs, and irony and briefly analysed their similarities and dif-

ferences. We divided the computational methods to neural and non-neural and discussed each group separately.

Chapter 3

Identification of Irony and Sarcasm

Overview. In this chapter, we will have a look at irony and sarcasm as the first type of non-literal language in our study and develop models to capture them in the context of the microblogging platform Twitter. We will first have a look at the literature on irony detection in the NLP community. In the ensuing section, we set up experiments to identify ironic comments in binary and multi-label scenarios. The binary classifier decides whether a tweet is ironic or not. The second model not only identifies irony but also categorises it into different identified types. A combination of sentiment, distributional semantic, and text surface features are used to develop these models. In the final part of the chapter, we will assess the models by analysing their performance in each task and finally conclude the chapter by summarising the achievements.

3.1 Introduction

In figurative language (also known as trope), there is a departure from the literal use of words. In order to decode meaning, therefore, it is not enough to rely solely on the literal sense of individual words. Irony and sarcasm

(2.2.2) are two types of such language that exploit this technique in similar ways. They "both involve deliberately saying something that is incongruous or the opposite of what the speaker knows to be true" (Hanks, 2013). This is sometimes formulated as a transgression of the Gricean maxim of quality (Grice, 1975)[1].

Under this assumption, it follows that the violation is only permissible thanks to shared knowledge between the speaker and the hearer. In order to achieve this goal, the speaker frames the message with some form of commentary or metamessage that signals the ironic or sarcastic nature of the message. This is usually realised through the negation of the original meaning (Haiman, 1998).

Regardless of their similarities, irony and sarcasm are not technically the same as they might be employed for different purposes. It is widely accepted that sarcasm involves some degree of verbal aggression and ridicule directed at the hearer, while irony can simply be used for humorous or emphatic effect. It has been shown that computational processing of irony and sarcasm requires some knowledge of the context in which they appear, sometimes including paralinguistic information (Wallace et al., 2014). In accordance with some of the published works in the literature, we will nonetheless disregard the differences between irony and sarcasm and as described in section 2.2.2, these phenomena will be treated as one.

Exploring irony has practical implications since the performance of sen-

[1] "Do not say what you believe to be false."

timent analysis systems is directly affected by knowledge about irony and sarcasm (Pozzi et al., 2016). This stresses the need to research ways to correctly identify and interpret them in running text.

To computationally analyse irony, we will have a look at two separate tasks of binary classification and multi-class classification in the context of social media. The particular platform of interest is Twitter and English-language tweets with its specific linguistic challenges involved (Van Hee et al., 2018a). In the binary classification scenario, the objective is to train a system that can label tweets as ironic or not. In the multi-class scenario, the idea is to label tweets with one of the four specified labels describing the type of irony (verbal irony by means of a polarity contrast, situational irony, other verbal irony, and non-ironic).

To tackle these problems, in this chapter, we describe two rich feature-based systems that address each of the tasks mentioned above. The proposed systems use a combination of sentiment, distributional semantic, and text surface features. The code and data used for the experiments in this chapter are freely available[2].

The rest of this chapter is organised as follows: Section 3.2 describes related work. Section 3.3 provides a comprehensive description of the overall methodology including pre-processing, feature representation, and system architecture. Sections 3.4 and 3.5 discuss experiments and results, Section 3.6 involves error analysis and finally Section 3.7 concludes the chapter with

[2]`https://github.com/omidrohanian/irony_detection`

some closing remarks.

3.2 Irony Detection in the Literature

There has been a recent surge of interest in the tasks of irony and sarcasm detection due in large part to the increasing popularity of social media and the availability of data from websites like Twitter and Reddit. Some recent work focus exclusively on irony or sarcasm in isolation (Joshi et al., 2016), under the assumption that sarcasm has a stronger impact on changing the sentiment of the overall message. However, in many cases, these terms are taken to be practically synonymous (Pozzi et al., 2016; Wallace et al., 2014; Ptáček et al., 2014). SemEval has a long-standing shared task on sentiment analysis that has also involved the processing of figurative language, including irony and sarcasm (Ghosh et al., 2015; Nakov et al., 2016). Results from recent shared tasks on sentiment analysis confirm that the top-performing teams increasingly employ deep learning methodologies, while classical machine learning models like SVM and logistic regression remain popular and can still perform on a similar level as state-of-the-art (Ghosh et al., 2015; Rosenthal et al., 2017).

3.3 Methodology: Dissecting Tweets

In what follows, we will describe the methodology used in the design of the supervised models which we employed for classification of irony. The novelty of this work lies in the particular manner we have constructed the feature set

and the novel way we dissect the structure of the tweets to understand and analyse their spatial structure. The models themselves are relatively simple in design. We used an ensemble soft voting classifier with logistic regression (section 2.3.1) and support vector machine (section 2.3.2) as component models, and create the feature sets using a combination of sentiment, semantic, and surface features. We leverage these handcrafted features in combination with dense vector representations which differ in details between binary and multi-class scenarios. The differences in feature engineering and representation between the two scenarios will be discussed in 3.3.2.

3.3.1 Pre-processing

For the experiments in this chapter, tweets were tokenised using NLTK's tweet tokeniser (Loper and Bird, 2002). Additional pre-processing was done to obtain a subset of the features that concerned surface orthographic information (e.g. all capitals, elongations, emoticons, etc.) and pattern-based named entities (e.g. time, place, user, etc.). For this, we used the ekphrasis toolkit (Baziotis et al., 2017). It employs an XML-based annotation scheme that made it easy to extract this information.

For sentiment features and embeddings, however, pre-processing beyond tokenisation was deemed unnecessary as emoji and word vectors we used were pre-trained on raw tweets.

3.3.2 Feature Representation

In our observation of English tweets, we noticed that they often follow a fairly consistent spatial pattern. Informative words are more likely to cluster at both ends of a tweet. Hashtags, while scattered throughout the whole text, tend to occur at the end. In ironic tweets, negative sentiments are more likely to be preceded by neutral or positive ones. An example is given in (1).

(1) What a golden morning. 😒

In order for the models to capture these spatial patterns and to provide a more rigorous representation of a tweet's structure, we propose the idea of decomposing a tweet into separate chunks and extracting features for each one separately. By concatenating these features, it is possible to partially preserve information about linear precedence. To this end, we simply split the sentences into two sections as represented in example (2) and (3).

(2) 8ams are just | so LOVELY .

surface features: *time*1 | *allcaps*2

(3) SEEING @AlpEmiel ON | SATURDAY whaddddddup #legend

surface features: *allcaps*1, *user*1 | *elongated*2, *hashtag*2

In examples (2) and (3), the numbers '1' and '2' signify the first and second sections of the tweet, respectively. We use the same split structure

for the representation of other features and pre-trained dense vectors.

Contrast is one of the essential characteristics of ironic language. One contribution of this work lies in the particular manner in which the notion of contrast is defined. Contrast is a marker of polarity shift and is usually seen as the presence of a positive sentiment referring to a negative situation, or vice versa (Riloff et al., 2013) which is sometimes referred to as "asymmetry of affect" (Clark and Gerrig, 1984).

Twitter language is non-standard and informal. Polarity shift can be realised through contrast between different elements of the tweet. The elements of a tweet are: text, hashtagged tokens, and emojis. We adopt a more inclusive stance with regards to the concept of contrast with the following scenarios:

1. Contrast between different parts of the same element of a tweet
    a. antithetical emojis
    b. antithetical hashtagged tokens
2. Contrast between two different elements of a tweet
    c. text and hashtagged tokens
    d. text and emojis
    e. hashtagged tokens and emojis

A sizable proportion of the tweets contain multiword hashtags, such as *#NotExcitedAboutThisAtAll* or *#goodluck*, that require segmentation. For

this we used ekphrasis' hashtag segmentation tool (Baziotis et al., 2017).

We separate the tweet and its segmented hashtagged tokens and run each group through the sentiment analysis tool from Stanford CoreNLP (Manning et al., 2014). CoreNLP assigns to an input any of the 5 sentiment classes from very negative to very positive (0 to 4). If the resulting hashtag and text scores are on opposite sides of this spectrum, we consider this as contrast type c. as defined in 2.

For d. and e. we follow a similar procedure. To approximate the sentiments present in emoji tokens, we use Emoji Sentiment Ranking Kralj Novak et al. (2015). This is a lexicon of 751 emojis whose sentiments are ranked based on the human annotation of 70,000 tweets in 13 European languages.

The resulting contrast feature is a binary value that is set to True if any one of the aforementioned forms of contrast is present in the tweet.

Relying on sentiment information from CoreNLP, we define an additional binary feature named Intensity. It checks whether the sentiment in a segment of the tweet is sharply positive/negative. This translates to a value of 0 or 4 in the sentiment scores for that particular segment. The rationale behind the definition of this feature is that too much of a positive emotion can, in specific contexts, imply a negative sentiment. To a lesser extent, the opposite is also true of an excessively negative emotion.

To track the changes of sentiment expressed throughout the whole tweet, we define sentiment patterns of Rise (R), Fall (F), and Stable (S) on a word-by-word basis and encode this information in a vector representing the num-

ber of S, R, F, RF, and FR patterns. For these features, we rely on information from Vader sentiment lexicon Gilbert (2014).

For dense vectors, we use `word2vec` embeddings pre-trained on a large twitter corpus as described in Godin et al. (Godin et al., 2015). One limitation of these embeddings is that they do not contain information on emojis. Therefore we have to complement this resource with additional embeddings specially trained on emojis (Eisner et al., 2016).

3.3.3 Task-specific Selected Features

As mentioned earlier, in this chapter, we look at irony classification in two separate scenarios. These will be described below using the terms 'subtask A' and 'subtask B' which correspond to binary and multi-class classification respectively. The dataset used in the experiments is from the Semeval 2018 shared task 3, which has two layers of annotation (binary and multi-label) that is specifically suited for these two subtasks.

3.3.3.1 Subtask A

For subtask A, we found that the best way to combine embeddings is through averaging, separately for left and right parts[3]. Features we combine with these vectors are the following: Surface features, Intensity (for left and right), and Contrast. A complete list of these features is available at Appendix A.

[3]Word and tweet embeddings are averaged independently, and subsequently the averages are concatenated.

3.3.3.2 Subtask B

For subtask B, concatenation of the embeddings was deemed more effective. Furthermore, we augment the combined embeddings with bigram tf-idf count vectors.

As a rhetorical trope, irony can often have subtle political and social dimensions, and is used frequently to express opinionated thoughts in general (Hutcheon, 1994). We noticed that adding topic modelling features to the developed system in subtask B slightly improves classification performance as these features can help the model capture more subtle forms of irony that tend to co-occur with certain topics and are not necessarily realised as polarity contrasts. Topic modelling of the tweets is done using Latent Dirichlet Allocation (LDA) (Blei et al., 2003) and Non-negative matrix factorisation (NMF) (Lee and Seung, 2001).

Other features we add to the above are: Surface features (we consider these with regard to both the whole tweet and its left and right splits), Intensity (for left and right), Contrast, and Vader-based Rise and Fall sentiment patterns.

3.4 Experimental Settings

The experiments are conducted using the data (text including emojis) provided in the shared task. Models were trained on the training set using 10-fold cross-validation, and predictions were made on the held-out test data.

	ironic	non-ironic	total
train	1911 (49.84%)	1923 (50.15%)	3834
test	311 (39.66%)	473 (60.33%)	784

Table 3.1: Statistics of the data for subtask A

Train and test data in both subtasks A and B are the same and only differ in their annotation. Tables 3.1 and 3.2 present the breakdown of the classes and the number of their instances in each subtask.

	non-ironic	clash	situational	other	total
train	1923 (50.15%)	1390 (36.25%)	316 (8.24%)	205 (5.34%)	3834
test	473 (60.33%)	164 (20.91%)	85 (10.84%)	62 (7.90%)	784

Table 3.2: Statistics of the data for subtask B

The most informative features are selected using recursive feature elimination (RFE) (Guyon et al., 2002). As a result, the algorithm uses 13 features for subtask A as listed in figure 3.1. They are concatenated with the vectors that were derived by separately averaging the words and emoji vectors of the left and right parts of tweets.

The best features derived from RFE for subtask B did not improve the performance of the model. Therefore we use all of the 87 features which are consequently augmented with the concatenation of the word and emoji vectors of tweets.

The baseline model included originally by the organisers of the Semeval

```
rightIntensity, contrast, date1, sad1, surprise1, url1,
date2, elongated2, laugh2, sad2, shocking2, url2, user2
```

Figure 3.1: The most informative features for subtask A

shared task is an SVM classifier which uses tf-idf feature vectors. We consider this as the benchmark and report the results for 2 different settings of our proposed model as follows:

- `setting 1`: the average of word and emoji vectors of bi-sectioned tweets
- `setting 2`: the concatenation of word and emoji vectors

In both settings, we combine vectors with best features and feed them to the classifiers. To achieve the best system for subtask A (`best system A`), we apply a voting classifier with soft voting between LR and SVM whose model components are based on `setting 1` plus the 13 best features that were selected using RFE for subtask A.

The best system for subtask B is a voting classifier between 3 LRs with 3 different class weights [4], as shown in Table 3.3. The components of the models are based on `setting 2` plus all features for subtask B.

[4]The 'situational' and 'other' categories are markedly underrepresented in the dataset (see figures in Table 3.2), and the class weights are chosen in order to put more emphasis on the minority classes.

	non-ironic	clash	situational	other
LR1	1	1	1	1
LR2	1	1	2	2
LR3	1	1	3	3

Table 3.3: Weights each LR classifier assigns to the 4 classes in subtask B

3.5 Results and Discussion

Splitting of the feature space into two parts was motivated by the observation that tweets are usually more informative around the beginning or the end where important words, hashtags and emojis cluster, and in many cases the tokens in the middle are less decisive in detecting irony. A recurrent pattern is when an idea or sentiment is expressed only to be negated towards the end. The split is an attempt to preserve this spatial structure.

Table 3.4 details the results for subtask A, and the results for subtask B are presented in Table 3.5 [5]. After cross-validation on the `TRAIN` set, the best system which is an ensemble voting classifier trained on models based on `setting 1 + best features of subtask A` achieves the highest record in F1-score and recall, but is outperformed in accuracy by its own component model. In terms of precision, it also scores lower than the system based on `setting 2 + all features`.

When tested on the `TEST` data, the best system for subtask A ranked third overall on the shared tasks' official leaderboard among 44 teams with

[5]For comparison with the best systems in the shared task, two of the top ranked models, namely 'THU NGN ' and 'NTUA-SLP' are also included for comparison with neural-based state-of-the-art

		Accuracy	Precision	Recall	F1-score
TRAIN	benchmark system	0.6375	0.6440	0.6096	0.6263
	LR with setting 1	0.6643	0.6543	0.6923	0.6728
	LR with setting 2	0.6502	0.6466	0.6578	0.6521
	LR with setting 1 + best features of subtask A	**0.6808**	0.6616	0.7357	0.6967
	LR with setting 2 + all features	0.6787	**0.6726**	0.6923	0.6823
	best system A	0.6742	0.6452	**0.7698**	**0.7020**
TEST	benchmark system	0.635	0.532	0.659	0.589
	random baseline	0.503	0.373	0.373	0.373
	best system A	0.6430 (15)	0.532 (20)	0.8360 (1)	0.6500 (3)
	THU NGN	0.735	0.630	0.801	0.705
	NTUA-SLP	0.732	0.654	0.691	0.672

Table 3.4: Results for subtask A

		Accuracy	Precision	Recall	F1-score
TRAIN	benchmark system	0.6064	0.4359	0.3540	0.3470
	LR with setting 1	0.6142	0.4952	0.3449	0.3278
	LR with setting 2	0.6239	**0.5394**	0.3796	0.3817
	LR with setting 1 + all features	0.6325	0.4867	0.3696	0.3550
	LR with setting 2 + all features	0.6450	0.5308	0.4061	0.4134
	best system B	**0.6458**	0.5280	**0.4122**	**0.4215**
TEST	benchmark system	0.569	0.416	0.364	0.341
	random baseline	0.416	0.241	0.241	0.241
	best system B	0.6709 (2)	0.4311 (11)	0.4149 (10)	0.4153 (9)
	THU NGN	0.605	0.486	0.541	0.495
	NTUA-SLP	0.652	0.496	0.512	0.496

Table 3.5: Results for subtask B

an F-score of 0.65. It has the second-highest score for recall. This indicates that the coverage of the model is extensive.

For subtask B, the best system is an ensemble voting classifier comprised of three logistic regression models based on `setting 2 + all features` with the set-up indicated in Table 3.3.

As can be seen in Table 3.5, it gives the best F1-score, accuracy, and recall when cross-validated on the `TRAIN` data. On the held-out `TEST` data, the system ranked 9th in terms of F1-score, and with 0.6709 accuracy ranked second out of all participating systems in the shared task.

	non-ironic	clash	situational	other
TRAIN	0.7064	0.6584	0.2768	0.0444
TEST	0.7652	0.4651	0.2595	0.0299

Table 3.6: Per-class F1-scores for the best system in subtask B

Table 3.6 shows the F1-scores for subtask B based on the system's performance on each individual label. In the case of irony by clash, the proposed system achieves an F1-score of 0.6584. This confirms that the features are informative enough to help the model capture this type of irony reasonably well, even though only 20.91% of the tweets belong to this class (Table 3.2).

However, in the case of situational irony, the system performs much worse compared to other categories. There are several possible factors that collectively contribute to this poorer performance. Situational irony is less studied in the literature and designing effective features to model it is more difficult.

By definition, it involves a situation that does not conform to the expectations of the speaker and elicits an emotional response (Shelley, 2001). Expectations differ among individuals, and people often react differently to the same events and stimuli, which further complicates the problem.

In the provided dataset, the number of instances of this type of irony is small (only 8.24% of the total in the `TRAIN` set), and there are no salient textual characteristics that can signal their occurrence while distinguishing them from irony by clash.

3.6 Error Analysis

Vast coverage in subtask A also means that the model is quick to judge a tweet as ironic, which translates to a large number of tweets getting tagged as 1. According to Table 3.1 the distribution of labels is slightly skewed towards non-ironic labels, but in predictions 62% of the tweets are tagged as ironic, which explains higher recall and lower precision (Table 3.4). This can be traced back to the inclusive definition of contrast as defined in 3.3.2.

The gold standard provided is not without faults. As an example, (4) is obviously an ironic tweet that is incorrectly labelled as 0 in the goldstandard[6]. Also in example (5) the word *tit* (altered in spelling for censorship), is being used in two ways; first in its literal sense, and the other to sarcastically refer to a politician as foolish. This was labelled as non-ironic in the dataset, which is subject to debate. The system correctly identified both of these

[6] *Corny* has a negative connotation, implying that the joke is unfunny and uninteresting

instances.

(4) Corny jokes are my absolute favorite

(5) #farage a t1t in public who doesnt agree with seeing t1ts in public #breastfeeding

Looking at the per-class performances in subtask B (Table 3.6), the best system is predicting non-ironic instances with a high F1-score of 0.7652. However, the F1-scores for other classes remain low.

The numbers for situational are lower than irony by clash, which seems logical because in order to effectively pinpoint a tweet as ironic by situation it is sometimes necessary to have access to information beyond the text which could involve a broader context (social, cultural, political, etc.) as exemplified in the following examples that are taken from the `TRAIN` set:

(6) Sure Staff... Now Hiring. http://t.co/HDgfxG7elF

(7) #mondaymorning pouring rain and i am singing 'the most wonderful time of the year' as i walk to the office

(8) Patrick Kielty hosting Radio 2's Comedy Awards...

In (6), textual information does not provide anything of significant value. If the user clicks on the link, it seems like the image is about an employment agency that is hiring. Normally, they supply staff to clients who are recruiting, but in this case, it is the agency itself which is recruiting, and this goes against expectation. Realisation of this instance as situational irony requires interpretation of the image, which in turn requires linking the name *Sure Staff* to an agency, and the background knowledge about the role of employment agencies.

Example (7) involves the interpretation of a rainy day on Monday morning as unpleasant, which is subjective. Example (8) implies that the comedian is not particularly known to be funny, which again requires background knowledge and is also dependent on the opinion of the annotator, as it could also read as a non-ironic sentence if the reader does not share the same impression of the comedian.

3.7 Summary

In this chapter, we have described supervised systems to identify ironic tweets and categorise them into separate types. The systems leveraging a combination of word/emoji vectors and features related to polarity contrast, intensity and text surface features achieved competitive results for binary classification of tweets as ironic/non-ironic. When compared with the results of the Semeval shared task 3, the proposed model is ranked third out of 44 participating systems due in large part to its high coverage in identifying ironic-

tweets.

For the subtask of multi-class classification, we have also used topic modelling features and features related to the distribution of polarity. The system is ranked ninth out of 32 participating systems with very competitive accuracy.

The experiments in this chapter showed that a spatial analysis of tweets results in better feature representation that helps classification models capture a wider percentage of ironic tweets. This particular feature engineering proved very successful in the binary scenario; however, it fell short of fully capturing the nuances among the different sub-types of irony. In particular, situational irony is a sub-type that was not sufficiently addressed due to the limitations of the linguistic context in resolving those cases. Observation of the dataset confirms that in cases where the tweet involves a URL, the contents of the external web page can play an important role in discriminating between ironic and non-ironic tweets. Therefore, the introduction of multimodal features is one possible future direction to enhance the performance of such models.

It could be argued that the expansion of the feature space into two parts automatically results in a richer feature representation and the improvements cannot necessarily be attributed to the preservation of the spatial order. During the course of the experiments, we generated a host of new features using polynomial combinations of the original features and the classification performance significantly decreased even after feature selection was applied. For

future work, however, a thorough contrastive analysis is needed to compare classification performance in scenarios where similar groups of features are generated per sentence/tweet, per region, and per word.

Chapter 4

Multiword Expressions I: Effect of Gaze Features

Overview. In this chapter, we start our study of MWEs and develop sequence tagging models to identify them in context. We will have a look at novel features from behavioural data and assess the results compared with models that only use linguistic information. In recent years gaze data has been increasingly used to improve and evaluate NLP models due to the fact that it carries information about the cognitive processing of linguistic phenomena. In this chapter, we conduct a preliminary study towards automatic identification of multiword expressions based on gaze features from native and non-native speakers of English. We report comparisons between a part-of-speech (POS) and frequency baseline with: i) a prediction model based solely on gaze data and ii) a combined model of gaze data, POS and frequency. Despite the challenging nature of the task, the best performance was achieved by the latter. Furthermore, we explore how the type of gaze data (from native versus non-native speakers) affects the prediction, showing that data from the two groups is discriminative to an equal degree. Finally, it is shown that late processing measures are more predictive than early ones, which is in line with previous research on idioms and other formulaic struc-

tures. Some parts of the literature review, annotation, and the final analysis in this work are done in joint collaboration with the authors listed Rohanian et al. (2017). The author is responsible for the feature selection, and the models and algorithms used in the experiments.

4.1 MWEs as Formulaic Language

In order to alleviate the burden that language comprehension poses on the short-term memory, the human brain uses frequently occurring formulaic sequences such as collocations and idioms, among others, and stores them as units in the long-term memory (Conklin and Schmitt, 2012). As a result of the efficacy of this approach, a large proportion of the spoken and written language is formulaic, with some corpus studies claiming that between 52% and 58% of the language in the analysed corpora falls into this category (Erman and Warren, 2000), and other studies claiming that this figure is around 32% (Foster, 2001). Given the frequency with which this phenomenon occurs, the automatic identification of formulaic language is of paramount importance for a number of Natural Language Processing (NLP) tasks and applications.

Conklin and Schmitt (2012) argue that our brains store and process very frequent and highly fixed combinations as "wholes" as opposed to single words being added together and that this difference in processing is reflected in eye-tracking data. Several eye-tracking studies discussed in Section 4.2 show that there is a processing advantage for formulaic sequences for both na-

tive and non-native speakers compared to controlled non-formulaic sequences. Based on this evidence, it could be hypothesised that the characteristics of formulaic language could be captured through differences in the gaze patterns between formulaic and non-formulaic sequences. In a similar way, gaze data has previously been successfully used in other NLP tasks such as part-of-speech tagging (Barrett et al., 2016) and evaluation of word embeddings (Søgaard, 2016), and it has been shown that gaze signals transfer across languages (Barrett et al., 2016). In this sense, automatically identifying formulaic sequences based on gaze features could not only contribute to potentially improving classification accuracy and gaining insight into the cognitive processing of such units, but can also provide a language-independent approach to the identification of formulaic phrases. However, it is important to note that almost all studies using gaze data to investigate formulaic language focus solely on idioms and that other types of formulaic units have been significantly understudied.

In this chapter, we conduct a preliminary study towards the identification of multiword expressions (MWEs) based on gaze features. An MWE (2.2.3) is commonly known as a combination of two or more words, not necessarily continuous, that pose difficulties on language processing (Sag et al., 2002) and that typically have syntactic and semantic idiosyncrasies (Fazly and Stevenson, 2006). In particular, we focus on two common types of MWEs, namely, Verb-Particle (e.g. *give up*) and Verb-Noun (e.g. *take place*) constructions.

We use the GECO corpus (Cop et al., 2016), a monolingual and bilingual

corpus of the eye-tracking data of participants reading a complete novel. The use of this data allowed a comparison between the gaze patterns of native and non-native English speakers, as well as a comparison of the predictive power of data obtained from these two groups for the present task. Furthermore, we explore a range of early and late measures of cognitive processing in order to determine which of the two groups of features carries more crucial linguistic information. In order to account for the fact that MWEs are often processed as unified structures, Conditional Random Fields (CRF) classifier was used to label sequences of words, together with a variety of early and late gaze features. As powerful discriminative models, CRFs are known to capture temporal correlations between the observations and consider the dependencies among possible labels, making them suitable for tasks where predicted labels are parts of a structure.

Contributions in this chapter are summarised as follows:

- We explore a novel approach to MWEs identification based on gaze data. We compare a POS + Frequency baseline with: i) a prediction model based solely on gaze features and ii) a prediction model based on gaze features, POS, and frequency.

- A comparison between the predictive power of gaze data from native and non-native speakers of English in the context of the current task.

- A comparison between the predictive power of several early and late gaze features in the context of the current task.

The code used in the experiments and the annotation of the MWEs are available at:

`https://github.com/omidrohanian/gaze-mwe-ranlp2017`.

The GECO corpus could be downloaded freely at:

`http://expsy.ugent.be/downloads/geco`.

4.2 Related Work

In this part, we present some related works from the fields of eye-tracking research and automatic identification of multiword units.

4.2.1 Eye Tracking and Formulaic Language

Eye-tracking is a process where an eye-tracking device measures the point of gaze of an eye (gaze fixation) or the motion of an eye (saccade) relative to the head and a computer screen (Duchowski, 2009). Fixations are eye movements which stabilise the retina over a stationary object of interest, which, in the case of reading research, is the written text and its units (letters, words, phrases, etc.). Gaze fixations and revisits (go-back fixations to a previously fixated object) have been widely used as measures of cognitive effort by taking into account their durations and the places in text where longer fixations occur (Duchowski, 2009). A series of studies on eye-tracking during reading show that gaze data is sensitive to phenomena such as word frequency, verb complexity and lexical ambiguity, as well as contextual effects on word perception (Rayner, 1975; Rayner and Duffy, 1986; Rayner, 2009;

Rayner et al., 2012).

Gaze data has been previously used to investigate formulaic language with the main focus on idiom research (Underwood et al., 2004; Siyanova-Chanturia et al., 2011; Conklin and Schmitt, 2012; Siyanova-Chanturia, 2013; Cutter et al., 2014; Carrol and Conklin, 2015). For example, Underwood et al. (2004) showed that native speakers read idioms faster and with fewer fixations compared to control non-idiomatic phrases and that the last word of the idiom was read faster than the last word in the control condition. Similarly, non-native readers produced fewer fixations when reading idioms than when reading control phrases, but there were no differences in the durations of those fixations (Underwood et al., 2004). Siyanova-Chanturia et al. (2011) corroborated the processing advantages of idioms over novel phrases and showed that idioms required less re-reading and less re-analysis. Interestingly, there were no significant differences in the early gaze measures, suggesting that early eye-tracking measures may not be suitable for investigation of formulaic language (Siyanova-Chanturia et al., 2011). This result may be explained with previous research on the predictability of single words showing strong effects in terms of shorter first fixation durations and a greater likelihood of skipping (Rayner and Well, 1996). However, Carrol and Conklin (2015) argue that this effect may not scale up to formulaic units in a simple fusion and suggest taking an approach balancing between local, lexical context and global discourse context. Assuming that the case of formulaic language is that "the whole is greater than the sum of the parts", Carrol and Conklin

(2015) suggest the use of a *hybrid* approach where formulaic language is analysed both as a whole and at the level of individual words. In order to partly account for this effect, we use an algorithm which represents the data as a sequence of words considering their neighbouring word features. The use of 'hybrid' here refers to the idea that formulaic language is processed both as a single unit and as a string of words with its own internal syntax. A computational model that takes into account both the dependencies between elements of MWEs and also considers the way they are processed by humans through skipping patterns (which are indicative of treating a string of words as one unit) can be said to follow such a hybrid strategy.

4.2.2 Identification of MWEs

MWEs have been investigated in computational linguistics based on their many different characteristics such as fixedness (Fazly and Stevenson, 2008; Constant and Fotopoulou, 2016), non-compositionality (Baldwin and Kim, 2010; Yazdani et al., 2015), and semi-productivity (Villavicencio, 2003; Vincze, 2012). We have used these properties as the main guidelines for annotating MWEs, specifically following the guidelines provided by the PARSEME project on identifying verbal MWEs.[1] High frequency of MWEs and in particular, the principle that MWEs usually are constructed from high-frequency word components have been studied extensively in computational linguistics (Granger and Meunier, 2008).

[1]https://typo.uni-konstanz.de/PARSEME/images/shared-task/guidelines/PARSEME-ST-annotation-guidelines-v6.pdf

In a recent MWE workshop (Savary et al., 2017), several language-independent systems have been proposed for identifying or extracting MWEs. When used in conjunction with CRF models (Scholivet and Ramisch, 2017) or structured perceptrons (Schneider et al., 2014a), Part-of-Speech (POS) tags have been shown to be useful features (especially when parsing information is not available) to identify MWEs. Schneider et al.'s (Schneider et al., 2014a) statistical sequence model has achieved the best F1-score of 60% in identifying all heterogeneous types of MWEs, which indeed shows how challenging the task is.

4.3 Eye Tracking Data

The GECO corpus (Cop et al., 2016) used in this study is, to the best of our knowledge, the most recent eye-tracking corpus for English, which: i) contains gaze data from a natural reading task (as opposed to e.g. single sentences), ii) is long enough to contain a sufficient number of MWEs, and iii) contains paired gaze data from native and non-native readers. Eye-tracking data was collected for both the English version of the novel and its translation in Dutch; however, in the current study, we only focus on the data about English.

The text of the corpus is a novel by Agatha Christie entitled "The Mysterious Affair at Styles", the English version of which contains 54,364 tokens and 5,012 unique types. The novel was selected based on the fact that its word frequency distribution was the most similar to the one in natural lan-

guage use, as observed in the Subtlex database (Cop et al., 2016). The novel was read by 14 English monolingual undergraduates from the University of Southampton and 19 Dutch (L1) - English (L2) bilingual students at Ghent University (intermediate and advanced). The two groups were matched on age and education level. The monolingual participants read only the English version of the novel, which amounted to a total of 5,031 sentences. The bilingual participants read chapters 1 - 7 in one language and 8 - 13 in the other in a counterbalanced order, thus reading 2,449 English sentences. The eight bilingual participants who read the first part of the novel in English read 2,852 English sentences.

The sampling rate of the eye tracking device was 1 kHz. Full details about the method and procedure used for the development of the corpus could be found in Cop et al. (2016).

4.4 Gaze Features

A number of gaze features were selected for the corpus and are listed in Table 4.1. All gaze features were averaged over 14 readers for the monolingual data and 19 readers for the bilingual data. We divided the features into *early* and *late* processing measures. Early measures capture processes such as lexical access and syntactic processing, as well as oculomotor processes and visual properties of the region. An example of such a measure is *first fixation duration* (Demberg and Keller, 2008). Late measures account for late syntactic processing, textual integration processes, lexical and syntac-

tic/semantic processing and disambiguation in general. An example of a late measure is the *total reading time* of a region, which is the sum of all fixations on a region, including refixations of the region after it was left (Demberg and Keller, 2008).

Table 4.1: Categorised Gaze Features

Early	WORD_FIRST_FIXATION_DURATION WORD_FIRST_RUN_FIXATION_COUNT WORD_FIRST_RUN_FIXATION_% WORD_FIRST_FIXATION_VISITED_WORD_COUNT WORD_FIRST_FIX_PROGRESSIVE WORD_SKIP
Late	WORD_FIXATION_COUNT WORD_FIXATION_% WORD_RUN_COUNT WORD_GO_PAST_TIME WORD_SELECTIVE_GO_PAST_TIME WORD_TOTAL_READING_TIME WORD_TOTAL_READING_TIME_% WORD_SPILLOVER WORD_AVERAGE_FIX_PUPIL_SIZE WORD_SECOND_FIXATION_DURATION WORD_SECOND_RUN_FIXATION_COUNT WORD_SECOND_RUN_FIXATION_% WORD_SECOND_FIXATION_RUN WORD_THIRD_FIXATION_DURATION WORD_THIRD_RUN_FIXATION_COUNT WORD_THIRD_RUN_FIXATION_% WORD_THIRD_FIXATION_RUN WORD_LAST_FIXATION_DURATION WORD_LAST_FIXATION_RUN

4.5 Experiments

This section presents the annotation procedure, method, and setup used to conduct the experiments, as well as the definition of the baseline.

4.5.1 Annotation

Two annotators with a linguistic background labelled the GECO corpus for Verb + Noun and Verb + Particle constructions. We have considered cases where the components of an MWE can occur with at most three words in between. The kappa inter-annotator agreement is k = 0.7864. We have resolved the annotation differences by employing a third annotator to decide in cases of disagreement.

In order to prepare sequences to be trained by the CRF model, we extract all patterns of Verb + Noun and Verb + Prepositions (and Verb + a list of other particles such as *up, down, over, etc*) from the corpus, with at most three words between the components. MWEs are tagged using the IOB format based on the annotations. The (B) tag stands for words appearing at the beginning, (I) for words occurring inside, and (O) for words that are outside of an MWE (Sang, 2002). Verb+Noun and Verb+Particle patterns, with a window of one word before and one word after, are fed into the CRF model as input sequences. In total, there are 381 sequences that contain MWEs and $5,837$ which do not.

4.5.2 CRF-based Sequence Labelling

For the task of sequence labelling with sparse data, we use Conditional Random Fields (CRFs) as explained in section 2.3.4. CRFs are capable of relaxing the strong independence assumptions present in similar models like HMMs, which make them a suitable choice in a structured prediction task

where context is of importance (Lafferty et al., 2001).

We use Pycrfsuite[2] which is a freely available Python wrapper around the crfsuite toolkit[3]. For the training algorithm, we use Adaptive Regularisation Of Weight Vector (AROW) that is suitable for handling inherently noisy labels in the training set (Crammer et al., 2009).

In order to extract features for the CRF model, given each sequence:

1. gaze features of each word in the sequence are added;

2. for the verb part of the sequence, we also add the features of the last component of the pattern (Verb + Noun or Verb + Particle);

3. for all other words of the sequence, on the other hand, we add the features of the verb component of the pattern.

The gaze features of the GECO corpus, used in this study are listed in Table 4.1.

Algorithm 4.1 Bootstrap aggregating on CRF labels

procedure BAGGING
  $N = MWE \cup \overline{MWE}$
  $[] \leftarrow result$
  $n_{test} \leftarrow \frac{1}{5}\|MW\|$
  $n_{train} \leftarrow \frac{4}{5}\|MW\|$
  $n_{models} \leftarrow \frac{\|N\|-2\times n_{test}}{2\times n_{train}}$
  for 100 times **do**
    $TEST \leftarrow \{sample\, of\, size\ n_{test}\ from\ MWE\} \cup \{sample\, of\, size\ n_{test}\ from\ \overline{MWE}\}$
    for $i = 1$ to n_{models} **do**
      $TRAIN \leftarrow \{sample\, of\, size\ n_{train}\ from\ (\overline{MWE} - TEST)\} \cup \{MWE - TEST\}$
      $C_i \leftarrow CRF(train, test)$
    $C^* \leftarrow mode\{C_1, C_2, ..., C_{n_{models}}\}$
    $result.add(Eval(C^*))$
  return $mean(result), std(result)$

[2] `https://python-crfsuite.readthedocs.io/en/latest/`

[3] `http://www.chokkan.org/software/crfsuite/`

4.5.3 Setup

In order to tackle the imbalance of data, we employ a bootstrap aggregating strategy (Breiman, 1996). We first randomly select one-fifth of the MWEs and the same number from non-MWEs as the test data. Then, we divide the remaining non-MWEs into several different sections with the same size as the remaining MWEs. We train the model on each section of non-MWEs and the whole training set of MWEs. We test the model on the held-out test data by obtaining the majority votes of different training models over the test sample. This process is performed 100 times, and the average and standard deviations of the precision, recall and F1-score measures are reported. The formalised approach is presented in Algorithm 1.

4.5.4 Baseline

We apply the same CRF and aggregating approach only with lexical features as the baseline. POS and word frequency are used as features. The GECO data is provided with the POS tags for the words, while word frequencies from the BNC corpus (Leech, 1992) are employed.

In the case of these lexical features, given each word feature f_w present in the input sequence, contextual features f_{w-1} and f_{w+1} are automatically retrieved and added to the feature set. This informs the model of what is happening in the immediate neighbourhood of each word in the sequence.

4.6 Results

The results of CRF labelling using different sets of features are reported, including POS tags, Frequency (referred to as FREQ), Early and Late gaze measures (Table 4.2).

Since most of the data are not MWEs and are thus irrelevant to the task, we report the results exclusively for the words at the beginning of the MWEs (B-MWE) and other words occurring within and at the end of the expressions (I-MWE).

In table 4.2, we have first shown that augmenting the lexical features (POS and FREQ) with Gaze has slightly improved the performance ($F = 70.05$ for B-MWE and $F = 54.0$ for I-MWE) compared to the baseline ($F = 63.6$ for B-MWE and $F = 48.06$ for I-MWE). Although based on the reported standard deviation measures, adding Gaze features might not be helpful in some parts of the data, in general, the combination of lexical features and the gaze information outperforms the baseline model and the model that uses Gaze features alone (Early and Late) ($F = 53.06$ for B-MWE and $F = 27.97$ for I-MWE).

We also compare the performance of Early and Late features in identifying MWEs in the second part of the table. We note that Late features appear to be more discriminative than Early features in identifying MWEs. Although in the case of the B-MWE, the improvement over Early features is minimal, the difference is more contrastive for I-MWE. Also, the standard deviation

Table 4.2: The performance (%) and Standard Deviation (std) (%) of CRF labelling models using different sets of features.

Features		Precision (std)	Recall (std)	F1-score (std)
FREQ	B-MWE	46.92 (12.17)	27.59 (13.89)	32.53 (12.03)
	I-MWE	37.00 (14.18)	10.09 (7.06)	14.76 (8.62)
POS	B-MWE	59.14 (4.75)	63.34 (11.92)	60.05 (6.46)
	I-MWE	56.43 (5.44)	39.03 (8.59)	45.44 (6.05)
POS + FREQ	B-MWE	59.95 (3.54)	68.26 (7.96)	63.6 (4.45)
	I-MWE	55.19 (4.78)	43.16 (7.77)	48.06 (5.56)
Gaze features (Early and Late)	B-MWE	51.43 (3.19)	55.55 (9.2)	53.06 (5.22)
	I-MWE	37.43 (5.95)	22.97 (6.07)	27.97 (5.19)
POS + FREQ + Gaze	B-MWE	66.68 (3.36)	74.03 (5.45)	**70.05 (3.48)**
	I-MWE	59.08 (4.8)	50.03 (5.87)	**54.0 (4.41)**
Early features	B-MWE	51.77 (5.14)	55.28 (21.74)	51.02 (12.94)
	I-MWE	37.53 (19.25)	9.73 (10.41)	13.38 (11.7)
Late features	B-MWE	50.16 (3.22)	56.06 (9.41)	52.54 (5.11)
	I-MWE	38.07 (5.0)	21.23 (6.21)	26.8 (5.84)
POS + FREQ + Early features	B-MWE	66.54 (3.68)	74.45 (6.73)	70.11 (3.82)
	I-MWE	60.01 (4.53)	49.16 (5.67)	53.85 (4.1)
POS + FREQ + Late features	B-MWE	65.0 (3.43)	74.12 (5.96)	69.59 (3.69)
	I-MWE	58.85 (4.19)	50.23 (5.77)	53.93 (3.90)

for the model using Late features confirms its superior reliability. These improvements are observed using Early or Late features by themselves and not in conjunction with POS+FREQ.

Furthermore, we have conducted an experiment with gaze features extracted from non-native speakers of English as way to compare efficacy of such features in different settings. Since the results indicate no significant differences between the two groups, we report only F1-scores using different features in Table 4.3. The superior model performance when using late gaze features over early ones is clearly visible in this table.

Table 4.3: The performance (F1-score%) comparison between data from native (L1) and non-native (L2) speakers.

Features		L1	L2
Gaze	B-MWE	53.06 (5.22)	54.26 (4.8)
	I-MWE	27.97 (5.19)	26.66 (5.11)
POS + FREQ	B-MWE	70.05 (3.48)	69.66 (3.07)
+ Gaze	I-MWE	54.0 (4.41)	52.84 (3.89)
Early Gaze	B-MWE	51.02 (12.94)	51.69 (13.3)
	I-MWE	13.38 (11.07)	11.63 (11.66)
Late Gaze	B-MWE	52.54 (5.11)	54.95 (5.04)
	I-MWE	26.8 (5.84)	27.24 (5.78)

4.7 Discussion of the Results

We now proceed to discuss the results presented above with regards to: i) MWEs identification accuracy, ii) comparison between the predictive power of gaze data of native versus non-native speakers, and iii) the predictive power of early versus late gaze features.

In terms of identification accuracy for MWEs, the best performance was achieved by the model combining POS + Frequency + Gaze data for both the beginning of the MWE ($F = 70.05$) and the words occurring inside the MWE ($F = 54.0$). Even though gaze features on their own performed significantly worse than the baseline, the combined model of Gaze + Freq + POS outperformed the baseline and achieved a performance comparable to the state-of-the-art in the field (Section 4.2.2). The lower values of the standard deviations for the combined model for both B-MWE and I-MWE also show that it is more reliable than the baseline in its prediction over 100 iterations. Furthermore, the fact that gaze features improve the classification accuracy means that readers process these structures somewhat differently compared with non-MWE units.

We did not observe significant differences in model accuracy when running parallel models on the data from the native speakers and the one from the non-native speakers, which indicates that both data sets are discriminative to an equal extent. It is important to note that the non-native speakers were highly proficient in English and that this result may not be replicated

with gaze data from less proficient readers. From a practical perspective, this is important with regards to the type of eye-tracking corpora which could be used in similar experiments in the future. Since such resources are scarce and expensive to obtain, it is reassuring to know that data from non-native speakers could be used equally well to automatically identify MWEs. However, a more extensive analysis is necessary to understand how the features might differ in general between L1 and L2 and whether there are certain categories where this change is significant.

From a psycholinguistic perspective, however, this finding is not in line with previous research on the differences in gaze patterns between native and non-native speakers reading formulaic language (Section 4.2.1). One reason for this could be that previous research using gaze data to explore the processing of formulaic language has focused predominantly on idioms, while here we discuss a different group of MWEs. Another reason for this could be the different data sets used in these studies, and conclusive results can only be drawn if idiom research is performed using the GECO corpus or vice-versa.

Finally, much in line with previous studies (e.g. Siyanova-Chanturia (2013)), we observe that early gaze features are not useful metrics for investigating formulaic language. It is important to note that late features were more discriminative even without using the entire feature set; there were no significant differences in performance even when removing late features related to the third run and last runs ($F = 0.52$ for B-MWE and $F =$

0.23 for I-MWE). In the experiments, the late features were notably better at identifying the words inside the MWEs and we hypothesise that this effect could be due to the fact that given the pattern of Verb + Noun and Verb + Particle constructions, these were the disambiguation regions of the MWEs. Another possible explanation for the superiority of late features could be that mental processing of MWEs occurs after the fact, meaning, after the word is first encountered in reading. Therefore, early gaze features are not expected to contain much information about whether a particular sequence of tokens are MWEs or not.

Some of the limitations of this experiment are related to averaging of data from multiple participants and the fact that the newly-released GECO corpus (Cop et al., 2016) has not yet been studied in detail and thus it may contain inaccuracies yet to be spotted. We plan to address the first limitation by conducting a study where individual models are built for each individual participant. This would allow the analysis of individual differences and the effects they have on the robustness of the model. We chose to use the GECO corpus since it was the only corpus available which allowed comparison of gaze data from native versus non-native speakers. Nevertheless, it would be interesting to compare the current results on the GECO data to results from more established eye-tracking corpora such as the Dundee corpus (Kennedy et al., 2013) in order to further assess the validity of these findings.

4.8 Summary

In this chapter, we began to study MWEs from the point of view of formulaic structures. We devised sequence tagging models based on structured prediction. We presented preliminary research towards using gaze data to identify multiword expressions automatically. We showed that MWEs are indeed viewed differently, and that the best classification performance was achieved by a combined model of gaze features, and frequency and POS tags, which outperformed models based on frequency and POS only and on gaze features only. Furthermore, it was demonstrated that there was no statistically significant difference between the performance of models using gaze data from native versus highly proficient non-native speakers of English, suggesting that data from both reader groups could be used for similar tasks in the future. Finally, consistent with previous research in the field, we showed that late gaze features are better predictors of formulaic language. The novelty of the work presented here was in the inclusion of gaze as an informative behavioural feature. To the best of our knowledge, this is the first work integrating gaze data with MWE identification. The annotations generated as part of the experiments are also freely available to other researchers. In the next chapter, we continue to study MWEs and will try to develop novel architectures that surpass state-of-the-art while addressing long-standing problems in the literature.

Chapter 5

Multiword Expressions II: Addressing Discontinuity

Overview. In chapter 4, we started the computational treatment of MWEs and developed models to tag them in running text. Experiments showed that inclusion of gaze features could improve the learning model. However, in real life, gaze features might not easily come by, and MWEs are not always strictly formulaic. Therefore in this chapter, we take a look at a wider variety of MWEs, including the ones with flexibility and gappy structures, and we assume no outside data beside syntactic information and pre-trained representation.

In what follows, we will introduce a new method to tag MWEs in running text using a linguistically interpretable language-independent deep learning architecture. Similar to the previous chapter, we model MWEs as a sequence labelling problem, however, whereas we began with CRF as a structured prediction model, in this chapter we experiment with the design of a novel neural-based model and make use of a more sophisticated token-based embedding.

Furthermore, here we specifically target discontinuity, an under-explored aspect that poses a significant challenge to computational treatment of MWEs.

Two neural architectures are explored: Graph Convolutional Network (2.3.6) and multi-head self-attention (2.3.8). GCN leverages dependency parse information, and self-attention attends to long-range relations. We finally propose a combined model that integrates complementary information from both through a gating mechanism.

The experiments on a standard multilingual dataset for verbal MWEs show that the model outperforms the baselines not only in the case of discontinuous MWEs but also in overall F-score.

5.1 Discontinuity in the MWE literature

Multiword expressions (2.2.3) are linguistic units composed of more than one word whose meanings cannot be fully determined by the semantics of their components (Sag et al., 2002; Baldwin and Kim, 2010). As they are fraught with syntactic and semantic idiosyncrasies, their automatic identification remains a major challenge (Constant et al., 2017). Occurrences of discontinuous MWEs are particularly elusive as they involve relationships between non-adjacent tokens (e.g. ***put*** *one of the blue masks* ***on***).

While some previous studies disregard discontinuous MWEs (Legrand and Collobert, 2016), others stress the importance of factoring them in (Schneider et al., 2014b). Using a CRF-based and a transition-based approach respectively, Moreau et al. (2018) and Al Saied et al. (2017) try to capture discontinuous occurrences with help from dependency parse information. Previously explored neural MWE identification models (Gharbieh et al., 2017) suffer

from limitations in dealing with discontinuity, which can be attributed to their inherently sequential nature. More sophisticated architectures are yet to be investigated (Constant et al., 2017).

Graph convolutional neural networks (GCNs) (Kipf and Welling, 2017) and attention-based neural sequence labelling (Tan et al., 2018) are methodologies suited for modelling non-adjacent relations and are hence adapted to MWE identification in this study. Conventional GCN (Kipf and Welling, 2017) uses a global graph for the entire input. We modify it such that GCN filters convolve nodes of dependency parse tree on a per-sentence basis. Self-attention, on the other hand, learns representations by relating different parts of the same sequence. Each position in a sequence is linked to any other position with $O(1)$ operations, minimising maximum path (compared to RNN's $O(n)$) which facilitates gradient flow and makes it theoretically well-suited for learning long-range dependencies (Vaswani et al., 2017).

The difference in the two approaches motivates the attempt to incorporate them into a hybrid model with an eye to exploiting their individual strengths. Other studies that used related methods in sequence labelling include Marcheggiani and Titov (2017) (GCN), and Strubell et al. (2018) (self-attention) where similar approaches were applied to Semantic Role Labelling (SRL) in multi-task settings. In this chapter, we show for the first time, how GCNs can be successfully applied to MWE identification, especially to tackle discontinuous ones. Furthermore, we propose a novel architecture that integrates GCN with self-attention achieving state-of-the-art results.

5.2 Neural Architecture to Address Discontinuity

To specifically target discontinuity, we explore two mechanisms feeding into a Bi-LSTM (Figure 5.1):

1. A GCN layer to act as a syntactic n-gram detector

2. An attention mechanism to learn long-range dependencies.

5.2.1 Graph Convolution as Feature Extraction

Standard convolutional filters act as sequential n-gram detectors (Kim, 2014). Such filters might prove inadequate in modelling complex language units like discontinuous MWEs. One way to overcome this problem is to consider non-sequential relations by attending to syntactic information in parse trees through the application of GCNs.

GCN is defined as a directed multi-node graph $G(V, E)$ where $v_i \in V$ and $(v_i, r, v_j) \in E$ are entities (words) and edges (relations) respectively. By defining a vector x_v as the feature representation for the word v, the convolution equation in GCN can be defined as a non-linear activation function f and a filter W with a bias term b as:

$$c = f(\sum_{i \in r(v)} W x_i + b) \tag{5.1}$$

where $r(v)$ shows all words in relation with the given word v in a sentence, and c represents the output of the convolution. Following Kipf and Welling (2017)

and Schlichtkrull et al. (2017), we represent graph relations using adjacency matrices as mask filters for inputs. We derive associated words from the dependency parse tree of the target sentence. Since we are dealing with a sequence labelling task, there is an adjacency matrix representing relations among words (as nodes of the dependency graph) for each sentence. We define the sentence-level convolution operation with filter W_s and bias b_s as follows:

$$C_s = f(W_s X^T A + b_s) \tag{5.2}$$

where X, A, and C are representation of words, adjacency matrix, and the convolution output, all at the level of sentence. The above formalism considers only one relation type, while depending on the application, multiple relations can be defined.

Kipf and Welling (2017) construct separate adjacency matrices corresponding to each relation type and direction. Given the variety of dependency relations in a parse tree and per-sentence adjacency matrices, we would end up with an over-parametrised model in a sequence labelling task. In this work, we simply treat all relations equally, but consider only three types of relations: 1) the head to the dependents, 2) the dependents to the head, and 3) each word to itself (self-loops). The final output is obtained by aggregating the outputs from the three relations.

5.2.2 Self-Attention

Attention (Bahdanau et al., 2014) helps a model address the most relevant parts of a sequence through weighting. As attention is designed to capture dependencies in a sequence regardless of distance, it is complementary to RNN or CNN where longer distances pose a challenge. In this work, we employ multi-head self-attention with a weighting function based on scaled dot product.

Based on the formulation of Transformer by Vaswani et al. (2017), in the encoding module an input vector x is mapped to three equally sized matrices K, Q, and V (representing key, query and value) and the output weight matrix is then computed as follows:

$$\text{Att}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d}})V \tag{5.3}$$

The timing signal required for the self-attention to work is already contained in the preceding CNN layers alleviating the need for position encoding.

5.2.3 Model Architecture

The overall scheme of the proposed model, composed of two parallel branches, is depicted in Figure 5.1. We employ multi-channel CNNs as the step preceding self-attention. One channel is comprised of two stacked 1D CNNs, and the other is a single 1D CNN. After concatenation and batch normalisation, a multi-head self attention mechanism is applied (Section 5.2.2).

Parallel to the self-attention branch, GCN learns a separate representa-

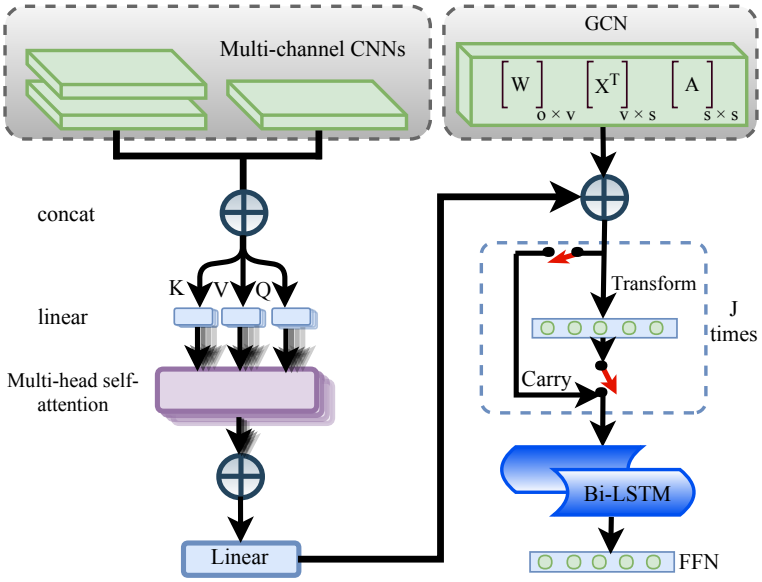

Figure 5.1: A hybrid sequence labelling approach integrating GCN (o: output dimension; v: word vectors dimension; s: sentence length) and Self-Attention.

tion (Section 5.2.1). The GCN layer is syntactically informed, and it retains crucial structural information that can translate to a better performance in MWE-based evaluation. As graph convolution is sensitive to the positional information from the syntax tree, we regard it is as a position-based approach.

On the other hand, the self-attention layer is intended to capture long-range dependencies in a sentence. It relates elements of the same input through a similarity measure irrespective of their distance. We, therefore, regard it as a content-based approach. As these layers represent different methodologies, we seek to introduce a model that combines their complementary traits in our particular task [1].

[1]Information from dependency parsing is helpful in informing the model of the phrasal structure of a sentence and the potential regions where MWEs could occur. However, not all these informative relations are recorded by a parser (e.g. refer to Figure 5.4) and to overcome this shortcoming, self-attention, which is a content-based method not sensitive to structural information is also included in the architecture.

Gating Mechanism. Due to the considerable overlap between the GCN and self-attention layers, a naive concatenation introduces redundancy which significantly lowers the learning power of the model. To effectively integrate the information, we design a simple gating mechanism using feed-forward highway layers (Srivastava et al., 2015) which learn to regulate information flow in consecutive training epochs. Each highway layer consists of a Carry (Cr) and a Transform (Tr) gate which decide how much information should pass or be modified. For simplicity Cr is defined as $1-Tr$. We apply a block of J stacked highway layers. Each layer regulates its input x using the two gates and a feedforward layer H as follows:

$$y = Tr \odot H + (1 - Tr) \odot x \tag{5.4}$$

where $\odot$ denotes the Hadamard product and Tr is defined as $\sigma(W_{Tr}x + b_{Tr})$. We set b_{Tr} to a negative number to reinforce carry behavior which helps the model learn temporal dependencies early in the training.

5.3 Experiments

Data. We experiment with datasets from the shared task on automatic identification of verbal MWEs (Ramisch et al., 2018). We focus on annotated corpora of four languages: French (FR), German (DE), English (EN), and Persian (FA) due to their variety in size and proportion of discontinuous MWEs. Tags in the datasets are converted to a variation of IOB which includes `B` (beginning of MWEs), `I` (other components of MWEs), and `O` (to-

kens outside MWEs), with the addition of `G` for arbitrary tokens in between the MWE components e.g. $make_{\texttt{[B]}}$ $important_{\texttt{[G]}}$ $decisions_{\texttt{[I]}}$.

ELMo. In our experiments, we make use of ELMo embeddings (Peters et al., 2018b) which are contextualised and token-based as opposed to type-based word representations like `word2vec` or `GLoVe` where each word type is assigned a single vector.

Validation. In the validation phase, we start with a strong baseline which is a CNN + Bi-LSTM model based on the top-performing system in the VMWE shared task (Taslimipoor and Rohanian, 2018). The implemented baseline differs in that we employ ELMo rather than `word2vec` resulting in a significant improvement. We perform hyper-parameter optimisation and make comparisons among the systems, including GCN + Bi-LSTM (GCN-based), CNN + attention + Bi-LSTM (Att-based), and their combination using a highway layer (H-combined) in Table 5.1.

5.4 Evaluation and Results

Systems are evaluated using two types of precision, recall and F-score measures: strict MWE-based scores (every component of an MWE should be correctly tagged to be considered as true positive), and token-based scores (a partial match between a predicted and a gold MWE would be considered as true positive). We report results for all MWEs as well as discontinuous ones specifically.

According to Table 5.1, GCN-based outperforms Att-based, and they

		All		Discontinuous			
		Token-based	MWE-based		MWE-based		
L	model	F	F	%	P	R	F
EN	baseline	41.37	35.38	32	24.44	10.48	14.67
	GCN-based	39.78	39.11		39.53	16.19	22.97
	Att-based	33.33	31.79		46.88	14.29	21.90
	H-combined	41.63	**40.76**		63.33	18.10	**28.15**
DE	baseline	62.27	57.17	43	69.50	45.37	54.90
	GCN-based	65.48	**61.17**		65.19	47.69	55.08
	Att-based	61.20	58.19		67.86	43.98	53.37
	H-combined	63.80	60.71		68.59	49.54	**57.53**
FR	baseline	76.62	72.16	43	75.27	52.04	61.54
	GCN-based	79.59	75.15		79.58	56.51	66.09
	Att-based	78.21	74.23		71.49	60.59	65.59
	H-combined	80.25	**76.56**		77.94	59.11	**67.23**
FA	baseline	88.45	86.50	14	67.76	55.88	61.29
	GCN-based	87.78	86.42		78.72	54.41	64.35
	Att-based	87.55	84.20		62.32	63.24	62.77
	H-combined	88.76	**87.15**		75.44	63.24	**68.80**

Table 5.1: Model performance (P, R and F) for development sets for all MWE and only discontinuous ones (%: proportion of discontinuous MWES)

both outperform the strong baseline in terms of MWE-based F-score in three out of four languages. Combining GCN with attention using highway networks results in further improvements for EN, FR and FA. The H-combined model consistently exceeds the baseline for all languages. As can be seen in Table 5.1, GCN and H-combined models each show significant improvement with regard to discontinuous MWEs, regardless of the proportion of such expressions.

In Table 5.2 we show the superior performance of the top systems on the test data compared to the previous state-of-the-art, ATILF-LLF (Al Saied et al., 2017) and SHOMA (Taslimipoor and Rohanian, 2018) in terms of

	All | Discontinuous			
	EN	DE	FR	FA
baseline	33.01 | 16.53	54.12 | 53.94	67.66 | 58.70	**81.62** | 61.73
GCN-based	36.27 | **24.15**	56.96 | 54.87	70.79 | 59.95	81.00 | **62.35**
H-combined	**41.91** | 22.73	**59.29** | **55.00**	**70.97** | **63.90**	80.04 | 61.90
ATILF-LLF	31.58 | 09.91	54.43 | 40.34	58.60 | 51.96	77.48 | 53.85
SHOMA	26.42 | 01.90	48.71 | 40.12	62.00 | 51.43	78.35 | 56.10

Table 5.2: Comparing the performance of the systems on Test data in terms of MWE-based F-score

MWE-based F-score. GCN works the best for discontinuous MWEs in EN and FA, while H-combined outperforms based on results for all MWEs except for FA.

Analysis. The overall results confirm our assumption that a hybrid architecture can mitigate errors of individual models and bolster their strengths. To demonstrate the effectiveness of the models in detecting discontinuous MWEs, in Figure 5.2, we plot their performance for FR and EN given a range of different gap sizes. As an ablation study, we show the results for the baseline, GCN-based, Att-based only, as well as H-combined models. GCN and Att-based models each individually outperform the baseline, and the combined model clearly improves the results further.

The example in Figure 5.3 taken from the English dataset demonstrates the way GCN considers relations between non-adjacent tokens in the sentence. The baseline is prone to disregarding these links. Similar cases captured by both GCN and H-combined (but not the baseline) are ***take*** *a final* ***look***, ***picked*** *one* ***up***, and ***cut*** *yourself* ***off***.

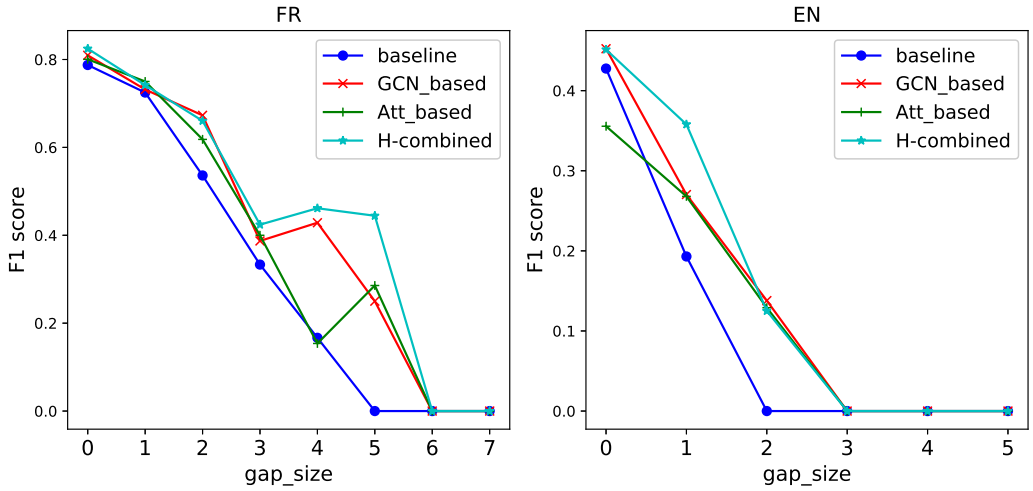

Figure 5.2: Model performance given different gap sizes

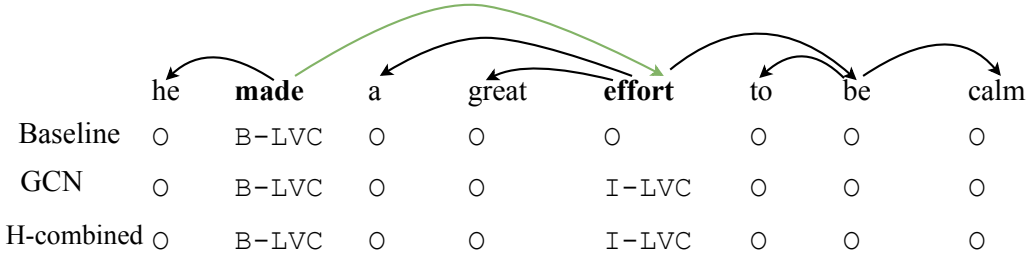

Figure 5.3: Sample sentence with a discontinuous MWE.

In more complicated constructs where syntactic dependencies might not directly link all constituents, GCN alone is not always conducive to optimal performance. In Figure 5.4, the sentence is in the passive form and MWE parts are separated by 5 tokens. This is an MWE skipped by GCN but entirely identified by H-combined model.

It is important to note that model performance is sensitive to factors such as percentage of seen expressions and variability of MWEs (Pasquer et al., 2018). In FA, 67% of the MWEs in the test set are seen at training time, making them easy to be captured by the baseline. Furthermore, 21%

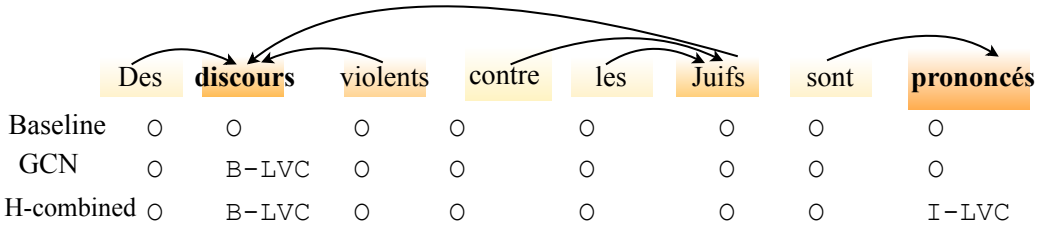

Figure 5.4: Example Sentence with a gappy occurrence. The intensity of the colouring corresponds to the attention weights assigned to each token.

of MWEs in FA and 15% in EN are discontinuous as opposed to 44% in FR and 38% in DE. Also in DE, a sizable portion of MWEs are verbal idioms (VIDs) which are known for their lexico-syntactic fixedness and prevalence of cranberry words. For such MWEs, the developed models compete with strong sequential baselines.

5.5 Summary

In this chapter, we introduced the application of GCN and attention mechanism to the identification of verbal MWEs and finally proposed and tested a hybrid approach integrating both models.[2] The particular point of interest was discontinuity in MWEs, which is an under-explored area. All the individual and combined models outperform state-of-the-art in all considered criteria and across several languages.

The GCN that we looked at receives its information from an adjacency matrix, whose relations are predetermined by a human. This conversion to adjacency is an inherently lossy process. We will examine a slightly different GCN in the next chapter (ch. 6) which is powered by information from a

[2]The code is available at `https://github.com/omidrohanian/gappy-mwes`.

self-attention mechanism, making the adjacency matrices richer.

So far, we have looked at two separate instances of non-literal language, namely, irony and MWEs. One of the research questions is to investigate whether information from one type of non-literal language would help identify another. In the following chapter, we will look at the interplay of metaphors and MWEs and whether we can develop an 'MWE-aware' model.

Chapter 6

Metaphor Processing and MWEs

Overview. Metaphor is a linguistic device in which a concept is expressed by mentioning another. Identifying metaphorical expressions, therefore, requires a non-compositional understanding of semantics. MWEs, on the other hand, are linguistic phenomena with varying degrees of semantic opacity and their identification poses a challenge to computational models. The interplay of metaphor and MWEs is an underexplored area. In Section 2.2.4 we saw how the two phenomena overlap, which provides the underpinning for the hypothesis that learning models can be enhanced by a knowledge of both when detecting instances of either one in running text. In this chapter, we analyse the interplay of metaphor and MWEs processing through the design of a neural architecture whereby classification of metaphors is enhanced by informing the model of the presence of MWEs. We will also present the first "MWE-aware" metaphor identification system paving the way for further experiments on the complex interactions of these phenomena. The results and analyses show that this proposed architecture outperforms the state-of-the-art on two different established metaphor datasets.

6.1 Introduction

Human language is rife with a wide range of techniques that facilitate communication and expand the capacities of thinking and argumentation. One phenomenon of such kind is metaphor (2.2.1). Metaphor is defined as a figure of speech in which the speaker makes an implicit comparison between seemingly unrelated things which nonetheless have certain common characteristics (Shutova, 2010). This is done to convey an idea which is otherwise difficult to express succinctly or simply for rhetorical effect.

As an example, in the sentence *she devoured his novels*, the verb *devour* is used in a metaphorical sense that implies reading quickly and eagerly. The literal and metaphorical senses share the element of intense desire which in turn helps to decode the meaning of the word in its context.

It is clear that a mere literal understanding of semantics would not result in a proper understanding of a metaphorical expression and a non-compositional approach would be required (Shutova et al., 2013; Vulchanova et al., 2019). The human brain is equipped with the necessary machinery to decode the intended message behind a metaphorical utterance. This involves mentally linking the seemingly unrelated concepts based on their similarities (Rapp et al., 2004).

Verbal MWEs (VMWEs) are another example of non-literal language in which multiple words form a single unit of meaning. These two phenomena are intersecting. Expressions like *take the bull by the horns*, *go places*, *kick the*

bucket, or *break someone's heart* can be categorised as metaphorical VMWEs. Based on this observation, we hypothesise that a metaphor classification model can be bolstered by knowledge of VMWEs.

In this work, we focus on how the identification of verbal metaphors can be helped by verbal MWEs. We devise a deep learning model based on attention-guided graph convolutional neural networks (GCNs) that encode syntactic dependencies alongside information about the existence of VMWEs and test the model on two established metaphor datasets.

6.2 Related Work

The tasks of MWE and metaphor identification share some similarities. Many idiomatic MWEs can be considered as lexicalised metaphors.

Idioms are where the overlap becomes clear (Kordoni, 2018). These are metaphors that have become set phrases and entered the lexicon because of overuse. It is important to note, however, that not all verbal metaphors are VMWEs. Metaphors that are less conventionalised and appear in creative context (e.g. within a poem or a literary piece) and are not established enough to make it as entries into dictionaries are examples of such cases. However, the distinction between these categories is not always clear, and few precise tests exist for the annotators to tell them apart (Gross, 1982). [1]

Most state-of-the-art MWE identification models are based on neural architectures (Ramisch et al., 2018; Taslimipoor and Rohanian, 2018) with

[1]See PARSEME annotation guidelines at `https://parsemefr.lis-lab.fr/parseme-st-guidelines/1.1/`

some employing graph-based methods to make use of structured information such as dependency parse trees (Waszczuk et al., 2019; Rohanian et al., 2019). Top-performing metaphor detection models also use neural methods (Rei et al., 2017; Gao et al., 2018), with some works utilising additional data such as sentiment and linguistic information to further improve performance (Mao et al., 2019; Dankers et al., 2019).

6.3 Graph Convolutional Networks

Graph Convolutional Networks (GCNs) (Kipf and Welling, 2016) are a variation of the classic CNNs that perform the convolution operation on nodes of a graph, making them suitable for capturing non-sequential inter-dependencies in the input.

Using the per-sentence formalism (Marcheggiani and Titov, 2017; Rohanian et al., 2019), GCN can be defined as:

$$GCN = f(WX^TA + b) \tag{6.1}$$

where W, X, A, b, and GCN refer to the weight matrix, representation of the input sentence, adjacency matrix, bias term, and the output of the convolution respectively. f is a nonlinearity which is often the relu function.

6.3.1 Multi-head Self-attention

Attention is a mechanism inspired by human visual attention which aims to encode sequences by emphasising their most informative parts through

weighting. Self-attention (Cheng et al., 2016), also referred to as intra-attention, is a special case of the attention mechanism which relates different parts of the same sequence and relies only on information from the same sequence. When the sequence is a series of words, this means encoding the sentence by learning correlations between words in the sentence. Self-attention is a powerful method to learn long-range dependencies in a sequence.

In this work, we use a particular form of self-attention introduced by Vaswani et al. (2017) in which the weighting is determined by scaled dot product. Given the input representation X, three smaller sized vectors are created. These are Query, Key, and Value which are represented with Q, K, and V respectively. The output of self-attention is computed with:

$$Att(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d}})V \tag{6.2}$$

N different self-attention mechanisms are activated in parallel. This approach is known as N-headed self-attention, where each head

$$H_i = Att(QW_i^Q, KW_i^K, V)$$

and the projections W_i^Q and W_i^K are parameter matrices. The outputs from these individual heads are later used in GCN layers Guo et al. (2019).

6.3.2 Attention Guided Adjacency

Central to GCN is the adjacency matrix where the relations between nodes are defined. Converting the graph of relations to an adjacency matrix involves a rule-based hard pruning strategy and potentially results in discard-

ing valuable information due to the sparsity of the matrix. Influenced by Guo et al. (2019), in this work we consider dependency parse information as an undirected graph with adjacency A. To obtain $\tilde{A}$, we combine matrix A with matrices H_0, H_1,..., H_{N-1} induced by the N-headed self-attention mechanism defined in Section 6.3.1.

Given an N-headed attention, each A is converted to several $\tilde{A}_i$s where $i \in \{1, 2, ..N\}$ and each $\tilde{A}_i$ is a linear combination of A and H_i.

$$\tilde{A}_i = \alpha \times H_i + (1 - \alpha) \times A \tag{6.3}$$

Each $\tilde{A}_i$ can be interpreted as a fully connected graph where a weight value determines the relation strength between every two nodes. In this case, a higher weight signifies a more substantial relation and a value close to zero would signal a lack of connection. These edge-weighted graphs are then fed to separate GCNs. A consolidated representation is finally achieved by a linear combination of the outputs from these N different GCNs.

The use of attention within the GCN network is motivated by the assumption that multi-hop paths between distantly related nodes could potentially be captured this way. We stack n layers of attention-guided GCNs using residual connections with n being a hyper-parameter that is tuned independently in each dataset.

Graph Attention (GAT) (Veličković et al., 2017) is a closely related work where the scope of attention is the neighbourhood of each node, whereas we make use of the entire sentence.

6.3.3 MWE-Aware GCN

In order to inform the model of the structural hierarchy within the sentence and encode information about MWEs, our attention-guided GCN component integrates information from two separate sources; namely, the dependency parse information and token-level relations between components of existing MWEs in the sentence. These correspond to adjacencies $\tilde{A}_{DEP}$ and $\tilde{A}_{MWE}$ which are fed each into separate GCNs, and the output is a concatenation of the outputs from both components:

$$GCN = concat[GCN_{s_{MWE}}; GCN_{s_{DEP}}] \tag{6.4}$$

6.4 Experiments

We describe the datasets used in the experiments and then provide details of the overall system.

6.4.1 Datasets

We apply the systems on two different metaphor datasets: MOH-X, and TroFi, which contain annotations for verb classification. Both of these datasets contain a set of sentences in which a single verb token is labelled as metaphorical or not. There is also an index provided that specifies the location of the target token in the sentence.

MOH-X. MOH-X is based on earlier work by Mohammad et al. (2016). It consists of short 'example' sentences from WordNet (Fellbaum, 1998)[2]

[2]Examples are sentences after the gloss that show in-context usage

with labels for metaphorical verbs along with associated confidence scores. Shutova et al. (2016) created a subset of this dataset, referred to as MOH-X, and added annotations for each verb and its argument. This dataset has 214 unique verbs.

TroFi. Similar to MOH-X, TroFi (Birke and Sarkar, 2006) has annotations for target verbs in each sentence. It has a comparatively longer average sentence length with 28.3 words per sentence compared to MOH-X's 8.0. The sentences in TroFi are constructed from the Wall Street Journal Corpus (Charniak et al., 2000). There are only 50 unique target verbs in this dataset.

6.4.2 MWE Identification

We extract MWEs using the GCN-based system proposed by Rohanian et al. (2019). Since we are focusing on verbal metaphors in this study, we train the system on the PARSEME English dataset Ramisch et al. (2018), which is annotated for verbal MWEs. As a result, predicted MWE labels in our target datasets are IOB formatted, where B and I denote the *beginning* and *inside* tokens of an MWE and O signifies tokens not belonging to MWEs.

We encode the relations between components of MWEs in each sentence using an adjacency matrix. Tokens of a sentence are nodes of the adjacency matrix; edges exist between tokens of an MWE. Relation matrices are then fed to the attention guided system, as explained in Section 6.4.3.

The numbers of verbal MWEs in correlation with target verbs in metaphor datasets are shown in Table 6.1. As can be seen, almost 16% of metaphors

	TroFi	MOH-X
verbal metaphor	1627	315
MWE	257	77

Table 6.1: Number of predicted MWEs among target verbs.

	MOH-X				TroFi			
Models	Acc	P	R	F1	Acc	P	R	F1
Gao et al. (2018)	78.5	75.3	**84.3**	79.1	73.7	68.7	74.6	72.0
RNN-HG (Mao et al., 2019)	79.7	79.7	79.8	79.8	74.9	67.4	**77.8**	72.2
RNN-MHCA (Mao et al., 2019)	79.8	77.5	83.1	80.0	**75.2**	68.6	76.8	72.4
BERTBaseline	78.04	78.38	77.87	77.82	70.38	70.54	68.89	68.84
BERT+GCN	79.44	79.79	79.36	79.31	72.01	72.32	70.45	70.65
BERT+MWE-Aware GCN	**80.47**	**79.98**	80.40	**80.19**	73.45	**73.78**	71.81	**72.78**

Table 6.2: Performance of MWE-Aware GCN against baselines and state-of-the-art on MOH-X and TroFi

in TroFi and 24% of metaphors in MOH-X are automatically labelled as VMWEs. This provides a strong motivation for incorporating this information into the metaphor identification system.

6.4.3 System Description

For our experiments, we devise two strong baselines and compare them against our proposed model. All three systems are built on top of a pre-trained BERT architecture (Devlin et al., 2019).

The starting baseline (BERTBaseline) is vanilla pre-trained BERT with a classification layer added on top. The other two models (BERT+GCN and BERT+MWE-Aware GCN) are created by adding extra layers with trainable

parameters on top of the BERT model, augmenting its original structure. [3]

BERT+GCN is BERT plus an attention-guided GCN that uses dependency parse information. Finally, BERT+MWE-Aware GCN refers to the system that uses BERT along with the added MWE-aware GCN component that utilises both dependency and VMWE information as detailed in Section 6.3.3.

Adam (Kingma and Ba, 2014) is used for optimising the network; the learning rate is controlled with a linear warmup scheduler in which the rate decreases linearly after increasing during a warmup period. In all the models, given the verb index in the dataset[4], and before passing the token-level output of the GCN to the softmax layer, we slice the output tensor based on the provided index and only select for the representation of the token of interest and subsequently pass this sliced tensor to the classification layer.

6.5 Results

We report the results in terms of accuracy, precision, recall and F_1-score, macro averaged over the measures obtained from 10 fold cross-validation. As can be seen in Table 6.2, our proposed model outperforms the baselines and also surpasses state-of-the-art in terms of F_1-score and precision in both datasets. As a whole, the results obtained for the two datasets are more homogeneous across the four metrics compared to the previous state-of-the-

[3]For all the experiments we use the pre-trained BERT model, `bert-base-uncased`, from the transformers library (Wolf et al., 2019).

[4]An index specifies the location of the target token.

art.

We have performed t-tests on the distribution of the predicted labels produced by our own models. In TroFi, the computed pairwise p-values in the case of ('BERT+GCN' & 'BERTBaseline') and ('BERT+MWE-Aware GCN' & 'BERTBaseline') are 0.00033 and 0.000013, respectively. In MOH-X, the same corresponding p-values are 0.000018 and 0.000007. These values clearly reject the null hypothesis and demonstrate statistical significance against our baseline.

In order to have a fair comparison with the previous state-of-the-art, it is important to consider their architectures. Gao et al. (2018), which our model outperforms in most criteria across the two datasets, is a BiLSTM-based system that uses a combination of ELMo and GLoVe vectors for input representation. The two models by Mao et al. (2019) are more competitive, especially in accuracy and precision for the TroFi dataset. RNN-HG and RNN-MHCA are BiLSTM-based systems grounded in linguistic theories of Selectional Preference Violation (SPV) (Wilks, 1975) and Metaphor Identification Procedure (MIP) (Group, 2007), which are based on the semantic contrast between the metaphorical word and its context or between the literal and contextualised meanings of a target token. These two models also make use of contextualised embeddings.

6.6 Discussion

The more significant portion of annotated VMWEs in both datasets are figurative and thus provide a valuable signal to metaphoricity. TroFi proved to be more challenging as sentences can be as long as 118 tokens with several different VMWEs and only a single token of interest which could be labelled as literal. On the other hand, MOH-X is more focused and VMWEs, for the most part, coincide with the target verb.

A notable pattern in the results is when the baselines miss a metaphor, and the proposed model correctly identifies it due to the presence of a non-compositional VMWE. A typical example is given below where *tack together*, identified initially as an MWE, signals metaphoricity:[5]

(1) He **tacked** together some verses.

There are examples of sentences falsely classified by BERT+GCN as metaphorical which are correctly identified as not by BERT+MWE-Aware GCN. This shows that the model has picked up informative cues and general patterns. There are also metaphors missed by BERT+GCN that do not have explicitly tagged VMWEs, but the proposed model is still able to capture them. Example (2) is an instance of such case:

(2) The residents of this village **adhered** to Catholicism.

[5]Target tokens are boldfaced

Due to their correlation with metaphoricity, VMWE information equips the model with the ability to identify metaphorical usage, which is reflected in the superior precision scores. However, this correlation is not always definitive, and in certain cases where a VMWE is realised in its literal meaning, the model might incorrectly associate its presence with metaphor. The following two sentences from MOH-X are examples of false positives influenced by VMWEs. Here, *jam the brake* and *land in* are VMWEs with literal meanings which can be idiomatic in other contexts:

(3) The driver **jammed** the brake pedal to the floor.

(4) The ship **landed** in Pearl Harbor

There are only a few such cases in MOH-X; however, in TroFi, the problem is exacerbated by longer sentences with multiple target tokens. One possible remedy could be to not attend to all the tokens in each sentence but instead, look at a certain window around the target token. We did not explore this idea in this work as it would defeat the purpose of attention-guided GCNs, but are open to considering it in future in such a way that accuracy is improved without hurting the precision scores which are higher in both datasets than the previous state-of-the-art.

6.7 Summary

In this chapter we presented a neural model to classify metaphorical verbs in their sentential context using information from the dependency parse tree and annotations for verbal multiword expressions. To the best of our knowledge, this is the first MWE-aware metaphor identification model, that demonstrates how the knowledge of MWEs can enhance performance of a metaphor classification model. Experiments showed that the resulting system sets a new state-of-the-art in several criteria across two benchmark metaphor datasets.

For future work, we plan to add VMWE annotations to the VU Amsterdam Corpus (Steen, 2010) which is the largest metaphor dataset and extend the experiments using that resource. Directionality of edges did not result in improvement in the models in this work, however for future we plan to develop GCNs that incorporate edge typing, which would enable us to differentiate between different MWE types and dependency relations while comparing them against the current models.

Chapter 7

Conclusions

Overview. In this closing chapter, we will first summarise the achievements made through the course of the present research (Section 7.1) and subsequently in Section 7.2, we will have another look at the research questions and the manner in which they were approached. Like any other undertaking, this work has points of strength and weakness and is constrained by certain limitations. These will be discussed in Section 7.3. Finally, in Section 7.4 we examine some possible future directions and conclude the thesis by examining the position of this work and the wider possible applications in the field of CL/NLP.

7.1 Summary of the Achievements

This thesis was conceived in order to investigate computational methods to identify instances of non-literal language within the context of tagging and classification. Throughout the course of this research, we studied and devised models explicitly geared to these particular tasks. The special point of focus was Multiword Expressions (MWEs), a group of semantically and syntactically idiosyncratic linguistic units that demonstrate varying degrees

of non-compositionality. We have designed models to classify and tag them in context. We have utilised linguistic and behavioural (eye-tracking) features to enrich those models.

For tagging, we started with structured prediction in chapter 4 using CRF, and later experimented with sophisticated neural architectures in chapter 5. A major contribution in our work with regards to tagging MWEs was the original development of a sequence labelling model based on self-attention and syntax-based graph convolutional neural networks buttressed with pre-trained contextualised embeddings. This model targeted the issue of gappy MWEs and continues to be the state-of-the-art on multiple languages in the standard PARSEME dataset.

We have also looked at irony and sarcasm in the context of social media in chapter 3, and based on the idea of semantic and sentiment contrast in sarcastic comments, we developed classification models through hand-crafted features. The resulting model competed in a shared task comprising more than 40 contestants where it ranked 3rd on the binary classification task. Twitter language can be garbled, cryptic and broken and very much dependent on memes, emoticons and other visual stimuli. However, the proposed model seemed to perform well given all the challenges involved in this task.

In the final chapter of the thesis, we looked at metaphor as another example of figurative language. It is known that verbal MWEs have a close relation to metaphors. Many expressions as in 'take the bull by the horns' (i.e. address the most challenging part of a problem) or 'put all one's eggs

in one basket' (i.e. rely on one particular course of action) are metaphorical in nature (Savary et al., 2017). The interrelation between these two different phenomena is an understudied area. We addressed this question by developing a classification model capable of identifying metaphorical expressions with help from information contained in MWEs. This attempt is the first MWE-aware metaphor identification system that reached and surpassed state-of-the-art on two established metaphor datasets.

To summarise, in this thesis, we devised several different experiments to study three closely related instances of non-literal language. The proposed models were tested on standard datasets and compared against strong baselines. In the course of the experiments, we designed two versions of GCNs and developed novel architectures that had not been previously tried on our targeted tasks. These contributions set new state-of-the-art on several tasks, created publicly available annotations for other researchers and culminated in hundreds of lines of open source code to make it possible for others to replicate the results.

7.2 Review of the Research Questions

In this section, we revisit the research questions and briefly discuss the ways they have been approached in this work.

Research Question 1. To what extent can state-of-the-art models identify non-literal language use in text?

Prior state-state-of-the-art, though impressive in overall F-score and ac-

curacy across many datasets, are lacking in sufficient generalisability in cases where patterns of non-literal language become more flexible and less predictable. For instance, in our experiments in Chapter 5 we showed how MWE identification models falter when dealing with discontinuous MWEs and unseen MWE types in test data. This stems from the inherent shortcoming of traditional machine learning models in handling non-contiguous spans and long-range dependencies.

In our work, we devised deep learning models with the aim to alleviate this shortcoming, specifically by employing better contextualised representations and by integrating methods like GCN and attention mechanism that can help a neural model capture indirect syntactic and semantic relations between elements in a sentence. These contributions resulted in the design of models that outperformed previous state-of-the-art, not only in specific criteria (e.g. generalisability to unseen data or discontinuous MWEs) but also in overall F-score, accuracy, and recall.

Fine-tuning giant pre-trained models like BERT have set very strong baselines for sentence and token classification tasks in NLP. In the case of verbal metaphor identification, we found that introduction of MWE features can easily reach or surpass state-of-the-art, when models were already powered by contextualised embeddings.

These tasks are far from solved, however, and there are still ongoing challenges involved in capturing certain types of non-literal language that depend on paralinguistic cues or suffer from inherent ambiguity. We will

mention these in Section 7.3.

Research Question 2. What are the differences and similarities in modelling different forms of non-literal language?

In Section 2.2.4 and 2.2.5, we examined the similarities and differences among the three types of non-literal language studied in this thesis. Our experiments show that all the three tasks benefited immensely from pre-trained word representations. Models of metaphor and MWE identification both seemed to benefit from syntactic information from dependency parse trees. Irony, on the other hand, seemed closer to sentiment tasks as evidenced by the improvement of models when coupled with sentiment features. Syntactic information could not be reliably integrated into an irony detection model because of the non-standard nature of Twitter language.

Irony, in particular, seemed to be more sensitive to spatial information and in order to disambiguate ironic from non-ironic instances, the entirety of an utterance (which can be more than a single sentence, for example in the case of a tweet) should be taken into account. A simple marker such as a hashtag, a misspelling, or repetition can change the content of a message. In verbal metaphor and MWE identification, however, decisions are mostly made at a local level spanning only a few tokens (e.g. gaps rarely stretch outside a window of 5 tokens) and, therefore, it is less often necessary to look beyond the immediate linguistic context.

Irony is perhaps the most creative and less rigid form of non-literal language out of the three in question. Metaphor is less conventionalised than

MWEs and can be harder to predict than irony on Twitter since in the datasets that we explored, the language is formal and polished, without all the markers present in social media.

What distinguishes MWE identification from the other two tasks is the tendency for the learning models to overfit on recurring MWE types and not generalise well to unseen data. Also, a major issue present in tagging MWEs is the phenomenon of discontinuity which was not an issue in the two other tasks we considered.

Research Question 3. To what extent can representation learning improve identification of non-literal text?

We tried three different types of representation learning in our experiments. We have used pre-trained type-based embeddings (`word2vec`), feature-based contextualised embeddings (using ELMo), and fine-tuning based contextualised embeddings (using BERT). Contextualised embeddings have become the standard method to represent textual data and they have led to very strong baselines using simple methods.

We used ELMo in our study of discontinuity in MWEs. Compared to simple type-based word embeddings, ELMo resulted in better capturing of nuances which reflected in improved performance using the same model that previously employed `word2vec` (CNN + Bi-LSTM model in Section 5.3). This can be attributed to the token-based nature of contextualised embeddings which recognises the range of semantic behaviour a word type can demonstrate in different contexts.

In the same vein, our experiments with fine-tuning BERT in Chapter 6 resulted in an impressive performance on two metaphor datasets without any additional layers. The introduction of additional trainable layers and addition of MWE information did result in a slight improvement in the performance of the model.

Our conclusion is that contextualised embeddings have progressed to a point where they can do most of the heavy lifting in bringing a model close to state-of-the-art in terms of its performance. Improving upon such a baseline can be a challenge as the representation is already packed with a lot of semantic and syntactic information.

Research Question 4. Can features from different phenomena help identify one particular kind of non-literal language?

To approach this question, we looked at verbal MWEs and verbal metaphors and devised a metaphor identification model that used information from MWEs. We called this 'MWE-aware' metaphor identification. In our particular experiment, the task was framed as token classification where a verb in a sentence is identified as metaphorical or not.

The results showed a small improvement across two metaphor datasets [1]. This can be attributed to the overlap between the two verbal phenomena and the fact that many common idioms are metaphorical in nature. Conversely, when new metaphors appear, some of them become established by usage and

[1]We assume the gains will depend on the register, topic, and the style of the annotated datasets.

crystallise in the form of MWEs.

Our experiments with behavioural data in Chapter 4 also showed that information from eye movements captured using eye-tracking software are informative in identifying a certain class of formulaic MWEs. Multi-modal MWE identification was not the main focus of this thesis, but in so far as it confirms that features beyond linguistic context can improve detetcion of one particular form of non-literal language, it helped address the last research question.

7.3 Strengths and limitations

The key points of strength in this work can be summarised as follows:

- **Creative use of features (including behavioural, sentiment, syntactic, among others) and feature engineering** for representing figurative language. In our experiments, we have used a variety of different sources of information to represent data and set up experiments. In some cases, we have creatively utilised features that had not been tried before. For instance, in the experiments of Chapter 4, we used behavioural data to study the effects of gaze features in MWE identification. Eye-tracking data had not been used for this task before, and our experiments have opened new avenues for exploring the effects of these features in figurative language and potentially in other areas of NLP.

By the same token, the use of metaphor information in Chapter 6 is an important step towards the application of new features in the task of figurative language identification and our experiments are a starting point for a much wider range of possible exploration in this area.

In the Chapter 3, we used a variety of different information, from linguistic representation to sentiment and topic modelling features, to inform our models of the richness of ironic utterances. Here, we not only used a variety of existing features, but we employed our own feature engineering to define features like contrast in a distinctive and original way (3.3.2). Our feature engineering included a spatial breakdown and analysis of the structure of tweets which enriched our data representation.

- **Replicability, and creation of new resources in the form of code and annotated data**. For almost all the experiments that were conducted in the course of this research, the code and the data have been made available so as to let other researchers replicate or experiment further in the same direction. In most cases, we have used established and standard datasets. In cases where we added a new layer of annotation (as in Chapter 4), we have made the annotation publicly available.

- **The use of state-of-the-art methods and reliability of the architectures**. In this work, we employed state-of-the-art methodologies

and kept improving the architecture of our models in newer experiments as the field of NLP was concurrently progressing. For instance, we adopted the use of contextualised embeddings based on transformers and language model pre-training (e.g. BERT and ELMo). We employed architectures that have proven stable and reliable in a variety of tasks. For instance, in our experiments in Chapter 5, we designed a neural network whose overall structure (i.e. CNN-based front-end followed by RNN) had proven superior in an MWE shared task in which a simpler but architecturally similar model had been successfully tested using a standard dataset for a variety of languages. Closely related architectures have also since been used in other tasks (e.g. Taslimipoor et al. (2019), Taslimipoor et al. (2019), and Asgari et al. (2019)), proving the reliability of this approach.

For non-neural experiments we employed standard and time-tested models like SVM, and logistic regression. Overall, the techniques used are reliable and the models or their closely related derivatives have been widely tested in different scenarios outside the context of the present research. However, given the task-specific needs and constraints, we designed models that were adapted to the specific circumstances of our experiments. For instance, since we were interested in avoiding the pitfalls of sequential models, we made some modifications to the CNN-LSTM architecture in order to account for discontinuity, all the while following a conventional overall architecture.

- **Task-specific focus and linguistic interpretability.** When choosing the design of the models and the overall strategy to frame the problems in each of the experiments, we studied the datasets and tried to understand the particular challenges involved in each case. The task-specific considerations influenced our decisions in the way we engineered the models. For instance in Chapter 3, irony datasets used informal and unpolished language which we employed to our advantage. Instead of using syntactic features derived from automatic parsers trained on standard text (which would have resulted in poor performance), we used typographical features (e.g. repetition or use of punctuation) and pre-trained representations for emoticons. In the case of MWE datasets in Chapter 5, we saw how gappy and unseen MWEs can create challenges for regular classifiers and designed a linguistically justified model that performed convolution by considering the syntactic relations between elements in the sentence and also considered long-range dependencies by the use of self-attention. In the experiments in Chapter 6, our decision to incorporate information from MWEs in a metaphor identification model was inspired by the analysis of verbal metaphor and verbal MWE datasets and the observation that a clear overlap existed between the two phenomena and the annotated datasets reflected this. Another instance where our feature engineering is derived from observation of the actual data is in Chapter 3, where we saw how informative features tended to cluster in either end of tweets and that was a basis for our

decision to break up tweets into separate chunks and analyse them separately. We regard these task-specific considerations as a strong point in our research, as opposed to the design of mere generic models that would have disregarded challenges and peculiarities of each linguistic phenomenon.

In spite of these strong points, like any other research, we faced limitations and challenges and there were some areas that require improvement. In what follows we list the weaknesses, limitations, and challenges in this work:

- **Limited number of phenomena considered and the differences between them**. In this work, we only study three instances of figurative language, namely MWEs, irony, and metaphor. In the case of MWEs and metaphor, we only stick to their verbal category and ignore other possible varieties. In the case of irony, we assume it to be practically the same as sarcasm, which is debatable, but a common practice in NLP. One major difference between irony and the other two cases in question is that most available irony datasets use a different register and style as they are derived from micro-blogging websites, rather than standard formal written language. One exception is Khodak et al. (2017), which uses a more standard language and is larger in size but it is self-annotated by the posters, rather by independent annotators, making the labels less reliable.

  As discussed in Section 2.2.4, 2.2.5, and 7.2, there are enough common-

alities among these phenomena to motivate their study and analysis of possible interrelations. However, there are also certain differences which can limit the range of common architectures and methods that can work across the board for all the three of them.

Furthermore, there are many more examples of figurative language which are not discussed here and there is no guarantee that the methods that have worked for the studied cases here would carry over to the analysis of other cases.

- **Inconsistent gains and small datasets** In some cases the datasets we had to work with were small. This problem was more apparent in the study of metaphor where in one case we were limited to a dataset of around 600 hundred sentences in size, which would radically complicate application of certain neural-based architectures and resulted in less stable results and in some cases (as discussed in the error analysis in 6.6) we saw inconsistent gains and while the overall F-score and accuracy improved over the baseline, in case by case analysis we saw occasional deterioration compared to the baseline. Due to the small size of some the datasets, we believe a proper analysis would have to involve the use of larger datasets.

- **Shortage of multilingual datasets for figurative language**.

  In the experiments in Chapter 5, we were able to test our architecture on several languages which is a strong point proving the language in-

dependence of the model. However, in our experiments for irony and especially metaphor, we were constrained to English datasets which severely limit the scope of the research and the claims. In recent years irony datasets in other languages have started to appear, however, there is a lack of publicly available annotated metaphor datasets in low resource languages to this day.

This problem is exacerbated when we wanted to design experiments to study the effect of MWEs in identification of metaphor. We were not able to find a publicly available non-English metaphor dataset and therefore worked only with English. We hope this will change in near future so similar experiments can be extended to other languages.

7.4 Ideas for Future Work

There are several possible ways to continue in the same line of research within the study of figurative language. The most obvious one is augmenting the experiments by analysis of other types of figurative language and how they might relate to the ones already discussed. As an example, simile is another figure of speech that involves a direct comparison between two concepts. It is closely related to metaphor but differs in that it makes the comparison explicit by highlighting the similarities. Examples in English include expressions like 'as brave as a lion' or 'like the cat that got the cream'. As in the case of metaphor, many of these cases have evolved into set phrases that can be considered as MWEs. Similes are closely related to metaphors, and in

fact they are sometimes, especially in the works of Aristotle, considered a subtype of metaphor as most metaphors can be rewritten as similes and vice versa. (Sam and Catrinel, 2006).

Simile identification either involves a sentence-level binary task or is framed as a sequence labelling problem where certain spans of text are tagged to denote the *tenor*, and the *vehicle*, which are the original concept and the comparison to describe it, respectively (Liu et al., 2018). An experiment of this kind would make it feasible to set up joint learning with MWEs or metaphors or create MWE-aware models similar to Chapter 6.

Another way to extend the present research is to apply the models (or their slightly modified variations) on a larger number of datasets. For metaphor, the most relevant is the VU Amsterdam Corpus (Steen, 2010) which is suitable for sequence labelling and classification. It consists of around $200,000$ words taken from 4 different registers of text (academic, conversations, fiction, and news) within the BNC-baby corpus. Models in the Chapter 6 can seamlessly be applied to this dataset with only slight modifications needed. This would overcome the problem of small metaphor datasets discussed in Section 7.3. Other possible ideas include finding metaphor datasets in other languages and re-running the experiments to see if the assumptions about the interrelation of MWEs and metaphors hold water in other languages as well. Because of the PARSEME annotated datasets[2], MWE tags can be obtained for a variety of different languages. However, the challenge would be to find

[2] `https://parsemefr.lis-lab.fr/parseme-st-guidelines/1.2/`

reliably annotated metaphor datasets in resource-poor languages.

It would be interesting to see if eye tracking data can be beneficial for classification and tagging models in the case of other types of MWEs beyond fixed expressions. It would also be possible to use other behavioural data including electroencephalography (EEG) features which have recently been shown effective across several NLP tasks (Hollenstein et al., 2019). Whether or not behavioural data would be effective in the case of irony or metaphor is also a valid possible research question.

As a broader question, we would be interested to investigate the type of NLP tasks that would benefit from MWE information. In the case of metaphor, we were confined to the verbal category and the register used in both datasets were similar. We would like to know to what degree MWEs would help in other contexts (non-verbal metaphor and MWEs, or MWEs and some other downstream application). We could also investigate whether it is possible to train MWE classifiers and taggers using informal (e.g. Twitter or similar) language and apply those features to the irony detection used in Chapter 3.

7.5 Summary

To conclude, this thesis focused on automatic identification of a number of linguistic phenomena that could all be referred to using the umbrella term figurative language. We made original contributions to the way these phenomena can be modelled. We used gaze features for the first time for the

classification of MWEs, created the first 'MWE-aware' metaphor detection model, and conducted the first study that specifically tackled the issue of discontinuity in neural MWE identification. Our models employed sound and reliable machine learning techniques and proved remarkably effective across several tasks and datasets.

Bibliography

Al Saied, H., M. Constant, and M. Candito (2017). The ATILF-LLF system for Parseme shared task: a transition-based verbal multiword expression tagger. In *Proceedings of the 13th Workshop on Multiword Expressions (MWE 2017)*, Valencia, Spain, pp. 127–132. Association for Computational Linguistics.

Asgari, E., N. Poerner, A. McHardy, and M. Mofrad (2019). Deepprime2sec: Deep learning for protein secondary structure prediction from the primary sequences. *bioRxiv*, 705426.

Bahdanau, D., K. Cho, and Y. Bengio (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Baldick, C. (2001). *The Concise Oxford Dictionary of Literary Terms (Oxford Paperback Reference)*. Oxford University Press, USA.

Baldwin, T., C. Bannard, T. Tanaka, and D. Widdows (2003). An empirical model of multiword expression decomposability. In *Proceedings of the ACL 2003 workshop on Multiword expressions: analysis, acquisition and treatment-Volume 18*, pp. 89–96. Association for Computational Linguistics.

Baldwin, T. and S. N. Kim (2010). Multiword expressions. *Handbook of natural language processing 2*, 267–292.

Bamberg, B. (1983). What makes a text coherent? *College Composition and Communication 34*(4), 417–429.

Barrett, M., J. Bingel, F. Keller, and A. Søgaard (2016). Weakly supervised part-of-speech tagging using eye-tracking data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, Volume 2, pp. 579–584.

Barrett, M., F. Keller, and A. Søgaard (2016). Cross-lingual transfer of correlations between parts of speech and gaze features. In *26th International Conference on Computational Linguistics (coling)*.

Baziotis, C., N. Pelekis, and C. Doulkeridis (2017, August). Datastories at semeval-2017 task 4: Deep lstm with attention for message-level and topic-based sentiment analysis. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, Vancouver, Canada, pp. 747–754. Association for Computational Linguistics.

Bingel, J. and A. Søgaard (2017). Identifying beneficial task relations for multi-task learning in deep neural networks. *EACL 2017*, 164.

Birke, J. and A. Sarkar (2006). A clustering approach for nearly unsupervised recognition of nonliteral language. In *11th Conference of the European Chapter of the Association for Computational Linguistics*.

Black, M. et al. (1979). More about metaphor. *Metaphor and thought 2*, 19–41.

Blei, D. M., A. Y. Ng, and M. I. Jordan (2003). Latent dirichlet allocation. *Journal of machine Learning research 3*(Jan), 993–1022.

Breiman, L. (1996). Bagging predictors. *Machine learning 24*(2), 123–140.

Carlin, G. (2005). *Life is Worth Losing*. MPI Media Group.

Carrol, G. and K. Conklin (2015). Eye-tracking multi-word units: some methodological questions. *Journal of Eye Movement Research 7*(5).

Chandler, D. (2007). *Semiotics: the basics*. Routledge.

Charniak, E., D. Blaheta, N. Ge, K. Hall, J. Hale, and M. Johnson (2000). Bllip 1987-89 wsj corpus release 1. *Linguistic Data Consortium, Philadelphia 36*.

Cheng, J., L. Dong, and M. Lapata (2016). Long short-term memory-networks for machine reading. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 551–561.

Cho, K., B. van Merriënboer, D. Bahdanau, and Y. Bengio (2014). On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pp. 103–111.

Cho, K., B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Clark, H. H. and R. J. Gerrig (1984). On the pretense theory of irony. *Journal of Experimental Psychology: General*.

Collobert, R., J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa (2011). Natural language processing (almost) from scratch. *Journal of machine learning research 12*(Aug), 2493–2537.

Conklin, K. and N. Schmitt (2012). The processing of formulaic language. *Annual Review of Applied Linguistics 32*, 45–61.

Constant, M., G. Eryiğit, J. Monti, L. Van Der Plas, C. Ramisch, M. Rosner, and A. Todirascu (2017). Multiword expression processing: A survey. *Computational Linguistics 43*(4), 837–892.

Constant, M. and A. Fotopoulou (2016). A systematic study on the fixedness degree of verbal multiword expressions: application to modern greek and french. PARSEME 6th general meeting in Struga.

Cop, U., N. Dirix, D. Drieghe, and W. Duyck (2016). Presenting geco: An eyetracking corpus of monolingual and bilingual sentence reading. *Behavior research methods*, 1–14.

Cramer, J. S. (2002). The origins of logistic regression.

Crammer, K., A. Kulesza, and M. Dredze (2009). Adaptive regularization of weight vectors. In *Advances in neural information processing systems*, pp. 414–422.

Cui, Z., R. Ke, Z. Pu, and Y. Wang (2018). Deep bidirectional and unidirectional lstm recurrent neural network for network-wide traffic speed prediction. *arXiv preprint arXiv:1801.02143*.

Cutter, M. G., D. Drieghe, and S. Liversedge (2014). Preview benefit in english spaced compounds. *Experimental Psychology Learning Memory and Cognition 40*(6).

Dankers, V., M. Rei, M. Lewis, and E. Shutova (2019, November). Modelling the interplay of metaphor and emotion through multitask learning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China, pp. 2218–2229. Association for Computational Linguistics.

Demberg, V. and F. Keller (2008). Data from eye-tracking corpora as evidence for theories of syntactic processing complexity. *Cognition 109*(2), 193–210.

Devlin, J., M.-W. Chang, K. Lee, and K. Toutanova (2018). Bert: Pre-

training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Devlin, J., M.-W. Chang, K. Lee, and K. Toutanova (2019, June). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota, pp. 4171–4186. Association for Computational Linguistics.

Do Dinh, E.-L., S. Eger, and I. Gurevych (2018). Killing four birds with two stones: Multi-task learning for non-literal language detection. In *Proceedings of the 27th International Conference on Computational Linguistics*, pp. 1558–1569.

Duchowski, A. (2009, feb). *Eye Tracking Methodology: Theory and Practice* (second ed.). Springer.

Eisner, B., T. Rocktäschel, I. Augenstein, M. Bošnjak, and S. Riedel (2016). emoji2vec: Learning emoji representations from their description. In *Proceedings of The Fourth International Workshop on Natural Language Processing for Social Media*, pp. 48–54. Association for Computational Linguistics.

Elkan, C. (2008). Log-linear models and conditional random fields.

Erman, B. and B. Warren (2000). The idiom principle and the open choice

principle. *Text-Interdisciplinary Journal for the Study of Discourse 20*(1), 29–62.

Fazly, A. and S. Stevenson (2006). Automatically constructing a lexicon of verb phrase idiomatic combinations. In *In Proceedings of EACL-06*, pp. 337–344.

Fazly, A. and S. Stevenson (2007). Distinguishing subtypes of multiword expressions using linguistically-motivated statistical measures. In *Proceedings of the Workshop on A Broader Perspective on Multiword Expressions*, pp. 9–16.

Fazly, A. and S. Stevenson (2008). A distributional account of the semantics of multiword expressions. *Italian Journal of Linguistics 1*(20), 157–179.

Fellbaum, C. (1998). *WordNet: An Electronic Lexical Database*. Cambridge, MA: MIT Press.

Fergusson, F. (2019). *Aristotle's poetics*. Macmillan.

Foster, P. (2001). Rules and routines: A consideration of their role in the task-based language production of native and non-native speakers. *Researching pedagogic tasks: Second language learning, teaching, and testing*, 75–93.

Gao, G., E. Choi, Y. Choi, and L. Zettlemoyer (2018, October-November). Neural metaphor detection in context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium, pp. 607–613. Association for Computational Linguistics.

Gerrig, R. J. and R. W. Gibbs Jr (1988). Beyond the lexicon: Creativity in language production. *Metaphor and Symbol 3*(3), 1–19.

Gharbieh, W., V. Bhavsar, and P. Cook (2017). Deep learning models for multiword expression identification. In *Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (*SEM 2017)*, Vancouver, Canada., pp. 54–64. Association for Computational Linguistics.

Ghosh, A., G. Li, T. Veale, P. Rosso, E. Shutova, J. Barnden, and A. Reyes (2015). Semeval-2015 task 11: Sentiment analysis of figurative language in twitter. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pp. 470–478.

Ghosh, D., A. R. Fabbri, and S. Muresan (2018). Sarcasm analysis using conversation context. *Computational Linguistics 44*(4), 755–792.

Gilbert, C. H. E. (2014). Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the Eighth International Conference on Weblogs and Social Media (ICWSM-14)*.

Giora, R. (1997). Understanding figurative and literal language: The graded salience hypothesis. *Cognitive Linguistics (includes Cognitive Linguistic Bibliography) 8*(3), 183–206.

Glucksberg, S., M. S. McGlone, Y. Grodzinsky, and K. Amunts (2001). *Understanding figurative language: From metaphor to idioms*. Number 36. Oxford University Press on Demand.

Godin, F., B. Vandersmissen, W. De Neve, and R. Van de Walle (2015). Multimedia lab acl wnut ner shared task: Named entity recognition for twitter microposts using distributed word representations. In *Proceedings of the Workshop on Noisy User-generated Text*, pp. 146–153.

Goldberg, Y. (2017). Neural network methods for natural language processing. *Synthesis Lectures on Human Language Technologies 10*(1), 1–309.

Goldberg, Y. and O. Levy (2014). word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method. *arXiv preprint arXiv:1402.3722*.

Granger, S. and F. Meunier (2008). *Phraseology: an interdisciplinary perspective*. John Benjamins Publishing Company.

Graves, A. (2012). Long short-term memory. In *Supervised sequence labelling with recurrent neural networks*, pp. 37–45. Springer.

Grice, H. P. (1975). Logic and conversation. In P. Cole and J. L. Morgan (Eds.), *Syntax and Semantics: Vol. 3: Speech Acts*, pp. 41–58. New York: Academic Press.

Gross, M. (1982). Une classification des phrases figées du français. *Revue québécoise de linguistique 11*(2), 151–185.

Group, P. (2007). Mip: A method for identifying metaphorically used words in discourse. *Metaphor and Symbol 22*(1), 1–39.

Guo, Z., Y. Zhang, and W. Lu (2019, July). Attention guided graph convolutional networks for relation extraction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, pp. 241–251. Association for Computational Linguistics.

Guyon, I., J. Weston, S. Barnhill, and V. Vapnik (2002). Gene selection for cancer classification using support vector machines. *Machine learning 46*(1-3), 389–422.

Haiman, J. (1998). *Talk is cheap: Sarcasm, alienation, and the evolution of language*. Oxford University Press on Demand.

Hanks, P. (2013). *Lexical analysis: Norms and exploitations*. MIT Press.

Hochreiter, S. (1998). The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 6*(02), 107–116.

Hochreiter, S. and J. Schmidhuber (1997). Long short-term memory. *Neural computation 9*(8), 1735–1780.

Hollenstein, N., M. Barrett, M. Troendle, F. Bigiolli, N. Langer, and C. Zhang (2019). Advancing nlp with cognitive language processing signals. *arXiv preprint arXiv:1904.02682*.

Hutcheon, L. (1994). *Irony's edge: The theory and politics of irony*. Psychology Press.

James, G., D. Witten, T. Hastie, and R. Tibshirani (2013). *An introduction to statistical learning*, Volume 112. Springer.

Joshi, A., V. Tripathi, P. Bhattacharyya, M. Carman, M. Singh, J. Saraswati, and R. Shukla (2016). How challenging is sarcasm versus irony classification?: A study with a dataset from English literature. In *Proceedings of the Australasian Language Technology Association Workshop 2016*, pp. 123–127.

Kennedy, A., J. Pynte, W. S. Murray, and S.-A. Paul (2013). Frequency and predictability effects in the dundee corpus: An eye movement analysis. *The Quarterly Journal of Experimental Psychology 66*(3), 601–618.

Khodak, M., N. Saunshi, and K. Vodrahalli (2017). A large self-annotated corpus for sarcasm. *arXiv preprint arXiv:1704.05579*.

Kim, Y. (2014). Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1746–1751.

Kingma, D. P. and J. Ba (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kipf, T. N. and M. Welling (2016). Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Kipf, T. N. and M. Welling (2017). Semi-supervised classification with graph

convolutional networks. In *International Conference on Learning Representations (ICLR)*.

Kordoni, V. (2018, July). Beyond multiword expressions: Processing idioms and metaphors. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, Melbourne, Australia, pp. 15–16. Association for Computational Linguistics.

Kotsiantis, S. (2011). Feature selection for machine learning classification problems: a recent overview. *Artificial Intelligence Review 42*(1), 157–176.

Kralj Novak, P., J. Smailović, B. Sluban, and I. Mozetič (2015). Emoji sentiment ranking 1.0. Slovenian language resource repository CLARIN.SI.

Kreuz, R. J. and K. E. Link (2002). Asymmetries in the use of verbal irony. *Journal of Language and Social Psychology 21*(2), 127–143.

Kuhn, M. and K. Johnson (2019). *Feature engineering and selection: A practical approach for predictive models*. CRC Press.

Lafferty, J., A. McCallum, and F. Pereira (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the eighteenth international conference on machine learning, ICML*, Volume 1, pp. 282–289.

Lakoff, G. (1987). The death of dead metaphor. *Metaphor and Symbolic Activity 2*(2).

Lakoff, G. and M. Johnson (2008). *Metaphors we live by*. University of Chicago press.

Lazar, G. (1996). Using figurative language to expand students' vocabulary. *ELT journal 50*(1), 43–51.

LeCun, Y., Y. Bengio, et al. (1995). Convolutional networks for images, speech, and time series.

Lee, D. D. and H. S. Seung (2001). Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, pp. 556–562.

Leech, G. (1992). 100 million words of english: the british national corpus (bnc). *Language Research 28*(1), 1–13.

Legrand, J. and R. Collobert (2016). Phrase representations for multiword expressions. In *Proceedings of the 12th Workshop on Multiword Expressions (MWE 2016)*, Berlin, Germany.

Li, L. and C. Sporleder (2010). Using gaussian mixture models to detect figurative language in context. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 297–300. Association for Computational Linguistics.

Liberman, A. (2019). The last shot at american idioms. `https://blog.oup.`

com/2019/09/last-shot-american-idioms/, Last accessed on 2020-01-25.

Liu, L., X. Hu, W. Song, R. Fu, T. Liu, and G. Hu (2018). Neural multitask learning for simile recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 1543–1553.

Loper, E. and S. Bird (2002). Natural language processing toolkit. http://www.nltk.org/.

Manning, C., M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, and D. McClosky (2014). The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pp. 55–60.

Mao, R., C. Lin, and F. Guerin (2019, July). End-to-end sequential metaphor identification inspired by linguistic theories. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, pp. 3888–3898. Association for Computational Linguistics.

Marcheggiani, D. and I. Titov (2017). Encoding sentences with graph convolutional networks for semantic role labeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 1506–1515.

Melamud, O., J. Goldberger, and I. Dagan (2016). context2vec: Learning generic context embedding with bidirectional lstm. In *Proceedings of The*

20th SIGNLL Conference on Computational Natural Language Learning, pp. 51–61.

Mikolov, T., I. Sutskever, K. Chen, G. S. Corrado, and J. Dean (2013). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pp. 3111–3119.

Mohammad, S., E. Shutova, and P. Turney (2016). Metaphor as a medium for emotion: An empirical study. In *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics*, pp. 23–33.

Montgomery, M., A. Durant, N. Fabb, T. Furniss, and S. Mills (2007). *Ways of reading: Advanced reading skills for students of English literature*. Routledge.

Moreau, E., A. Alsulaimani, A. Maldonado, and C. Vogel (2018). CRF-Seq and CRF-DepTree at PARSEME Shared Task 2018: Detecting verbal MWEs using sequential and dependency-based approaches. In *Proceedings of the Joint Workshop on Linguistic Annotation, Multiword Expressions and Constructions (LAW-MWE-CxG-2018)*, pp. 241–247. Association for Computational Linguistics.

Nakov, P., A. Ritter, S. Rosenthal, F. Sebastiani, and V. Stoyanov (2016). Semeval-2016 task 4: Sentiment analysis in twitter. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pp. 1–18.

Norbury, C. F. and D. V. M. Bishop (2003). Narrative skills of children with communication impairments. *International Journal of Language & Communication Disorders 38*(3), 287–313.

Olah, C. and S. Carter (2017). Research debt. *Distill 2*(3), e5.

Oxford Learner's Dictionaries (2020a). figurative. `https://www.oxfordlearnersdictionaries.com/definition/english/figurative?q=figurative`, Last accessed on 2020-01-24.

Oxford Learner's Dictionaries (2020b). irony. `https://www.oxfordlearnersdictionaries.com/definition/english/irony?q=irony`, Last accessed on 2020-01-24.

Oxford Learner's Dictionaries (2020c). metaphor. `https://www.oxfordlearnersdictionaries.com/us/definition/english/metaphor?q=metaphor`, Last accessed on 2020-01-25.

Oxford Learner's Dictionaries (2020d). trope. `https://www.oxfordlearnersdictionaries.com/definition/english/trope?q=trope`, Last accessed on 2020-01-24.

Pasquer, C., A. Savary, J.-Y. Antoine, and C. Ramisch (2018). Towards a variability measure for multiword expressions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, Volume 2, pp. 426–432.

Pavlov, P. (2009). Visualizing anger. `https://www.behance.net/gallery/371697/Visualizing-Anger`, Last accessed on 2020-04-15.

Pawelec, Andrzej (2020). The death of metaphor. `https://filg.uj.edu.pl/documents/41616/4333138/12308-Pawelec.pdf`, Last accessed on 2020-01-24.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. (2011). Scikit-learn: Machine learning in python. *Journal of machine learning research 12*(Oct), 2825–2830.

Pennington, J., R. Socher, and C. Manning (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543.

Peters, M. E., M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer (2018a). Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Peters, M. E., M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer (2018b). Deep contextualized word representations. In *Proceedings of NAACL*.

Pilehvar, M. T. and J. Camacho-Collados (2020). *Embeddings in Natural Language Processing*. Morgan and Claypool.

Popa-Wyatt, M. (2017). Go figure: understanding figurative talk. *Philosophical Studies 174*(1), 1–12.

Pozzi, F. A., E. Fersini, E. Messina, and B. Liu (2016). *Sentiment analysis in social networks*. Morgan Kaufmann.

Ptáček, T., I. Habernal, and J. Hong (2014). Sarcasm detection on Czech and English twitter. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pp. 213–223.

Ramisch, C., S. Cordeiro, A. Savary, V. Vincze, V. Mititelu, A. Bhatia, M. Buljan, M. Candito, P. Gantar, V. Giouli, et al. (2018). Edition 1.1 of the PARSEME shared task on automatic identification of verbal multiword expressions. In *the Joint Workshop on Linguistic Annotation, Multiword Expressions and Constructions (LAW-MWE-CxG-2018)*, pp. 222–240.

Rapp, A. M., D. T. Leube, M. Erb, W. Grodd, and T. T. Kircher (2004). Neural correlates of metaphor processing. *Cognitive brain research 20*(3), 395–402.

Rayner, K. (1975). The perceptual span and peripheral cues in reading. *Cognitive Psychology 7*(1), 65–81.

Rayner, K. (2009). Eye movements and attention in reading, scene perception, and visual search. *The quarterly journal of experimental psychology 62*(8), 1457–1506.

Rayner, K. and S. A. Duffy (1986). Lexical complexity and fixation times in reading: Effects of word frequency, verb complexity, and lexical ambiguity. *Memory & Cognition 14*(3), 191–201.

Rayner, K., A. Pollatsek, J. Ashby, and C. Clifton Jr (2012). *Psychology of reading*. Psychology Press.

Rayner, K. and A. D. Well (1996). Effects of contextual constraint on eye movements in reading: A further examination. *Psychonomic Bulletin & Review 3*(4), 504–509.

Rei, M., L. Bulat, D. Kiela, and E. Shutova (2017). Grasping the finer point: A supervised similarity network for metaphor detection. *arXiv preprint arXiv:1709.00575*.

Reyes, A., P. Rosso, and T. Veale (2013). A multidimensional approach for detecting irony in twitter. *Language resources and evaluation 47*(1), 239–268.

Riloff, E., A. Qadir, P. Surve, L. De Silva, N. Gilbert, and R. Huang (2013). Sarcasm as contrast between a positive sentiment and negative situation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 704–714.

Ritchie, L. D. and V. Dyhouse (2008). Hair of the frog and other empty metaphors: The play element in figurative language. *Metaphor and Symbol 23*(2), 85–107.

Roberts, R. M. and R. J. Kreuz (1994). Why do people use figurative language? *Psychological Science 5*(3), 159–163.

Rohanian, O., S. Taslimipoor, S. Kouchaki, L. A. Ha, and R. Mitkov (2019). Bridging the gap: Attending to discontinuity in identification of multiword expressions. *arXiv preprint arXiv:1902.10667*.

Rohanian, O., S. Taslimipoor, V. Yaneva, and L. A. Ha (2017). Using gaze data to predict multiword expressions.

Rosenthal, S., N. Farra, and P. Nakov (2017). Semeval-2017 task 4: Sentiment analysis in twitter. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pp. 502–518.

Sag, I. A., T. Baldwin, F. Bond, A. Copestake, and D. Flickinger (2002). Multiword expressions: A pain in the neck for nlp. In *International conference on intelligent text processing and computational linguistics*, pp. 1–15. Springer.

Sam, G. and H. Catrinel (2006). On the relation between metaphor and simile: When comparison fails. *Mind & Language 21*(3), 360–378.

Sang, E. T. K. (2002). Introduction to the conll-2002 shared task: Language-independent named entity recognition.

Sanz, H., C. Valim, E. Vegas, J. M. Oller, and F. Reverter (2018). Svm-rfe: selection and visualization of the most relevant features through non-linear kernels. *BMC bioinformatics 19*(1), 1–18.

Savary, A., C. Ramisch, S. Cordeiro, F. Sangati, V. Vincze, B. Qasemizadeh, M. Candito, F. Cap, V. Giouli, I. Stoyanova, et al. (2017). The parseme shared task on automatic identification of verbal multiword expressions. In *Proceedings of the 13th Workshop on Multiword Expressions (MWE 2017)*, pp. 31–47.

Scheuneman, J., K. Gerritz, and S. Embretson (1991). Effects of prose complexity on achievement test item difficulty. *ETS Research Report Series 1991*(2), i–53.

Schlichtkrull, M., T. N. Kipf, P. Bloem, R. v. d. Berg, I. Titov, and M. Welling (2017). Modeling relational data with graph convolutional networks. *arXiv preprint arXiv:1703.06103*.

Schneider, N., E. Danchik, C. Dyer, and N. A. Smith (2014a). Discriminative lexical semantic segmentation with gaps: Running the mwe gamut. *Transactions of the Association for Computational Linguistics 2*, 193–206.

Schneider, N., E. Danchik, C. Dyer, and N. A. Smith (2014b). Discriminative lexical semantic segmentation with gaps: Running the MWE gamut. *TACL 2*, 193–206.

Scholivet, M. and C. Ramisch (2017, April). Identification of ambiguous multiword expressions using sequence models and lexical resources. In *Proceedings of the 13th Workshop on Multiword Expressions (MWE 2017)*, Valencia, Spain, pp. 167–175. Association for Computational Linguistics.

Sheinfux, L. H., T. A. Greshler, N. Melnik, and S. Wintner (2017). Verbal mwes: Idiomaticity and flexibility. *Representation and Parsing of Multiword Expressions*, 5–38.

Shelley, C. (2001). The bicoherence theory of situational irony. *Cognitive Science 25*(5), 775–818.

Shutova, E. (2010). Models of metaphor in nlp. In *Proceedings of the 48th annual meeting of the association for computational linguistics*, pp. 688–697. Association for Computational Linguistics.

Shutova, E., D. Kiela, and J. Maillard (2016). Black holes and white rabbits: Metaphor identification with visual features. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 160–170.

Shutova, E., S. Teufel, and A. Korhonen (2013). Statistical metaphor processing. *Computational Linguistics 39*(2), 301–353.

Sikos, L., S. W. Brown, A. E. Kim, L. A. Michaelis, and M. Palmer (2008). Figurative language:" meaning" is often more than just a sum of the parts. In *AAAI Fall Symposium: Biologically Inspired Cognitive Architectures*, pp. 180–185.

Siyanova-Chanturia, A. (2013). Eye-tracking and erps in multi-word expression research: A state-of-the-art review of the method and findings. *The Mental Lexicon 8*(2), 245–268.

Siyanova-Chanturia, A., K. Conklin, and N. Schmitt (2011). Adding more fuel to the fire: An eye-tracking study of idiom processing by native and non-native speakers. *Second Language Research 27*(2), 251–272.

Smith, N. A. (2011). Linguistic structure prediction. *Synthesis lectures on human language technologies 4*(2), 1–274.

Søgaard, A. (2016). Evaluating word embeddings with fmri and eye-tracking. *ACL 2016*, 116.

Srivastava, R. K., K. Greff, and J. Schmidhuber (2015). Highway networks. *arXiv preprint arXiv:1505.00387*.

Steen, G. (2010). *A method for linguistic metaphor identification: From MIP to MIPVU*, Volume 14. John Benjamins Publishing.

Strubell, E., P. Verga, D. Andor, D. Weiss, and A. McCallum (2018, October). Linguistically-Informed Self-Attention for Semantic Role Labeling. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Brussels, Belgium.

Strubell, E., P. Verga, D. Belanger, and A. McCallum (2017). Fast and accurate entity recognition with iterated dilated convolutions. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 2670–2680.

Sutskever, I. (2013). *Training recurrent neural networks*. University of Toronto Toronto, Ontario, Canada.

Sutskever, I., O. Vinyals, and Q. V. Le (2014). Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pp. 3104–3112.

Sutton, C., A. McCallum, et al. (2012). An introduction to conditional random fields. *Foundations and Trends® in Machine Learning 4*(4), 267–373.

Tan, Z., M. Wang, J. Xie, Y. Chen, and X. Shi (2018). Deep semantic role labeling with self-attention. In *AAAI Conference on Artificial Intelligence*.

Taslimipoor, S. and O. Rohanian (2018). SHOMA at Parseme shared task on automatic identification of VMWEs: Neural multiword expression tagging with high generalisation. *arXiv preprint arXiv:1809.03056*.

Taslimipoor, S., O. Rohanian, et al. (2019). Cross-lingual transfer learning and multitask learning for capturing multiword expressions. In *Proceedings of the Joint Workshop on Multiword Expressions and WordNet (MWE-WN 2019)*, pp. 155–161.

Taslimipoor, S., O. Rohanian, and S. Može (2019). Gcn-sem at semeval-2019 task 1: Semantic parsing using graph convolutional and recurrent neural networks. Association for Computational Linguistics.

Underwood, G., N. Schmitt, and A. Galpin (2004). The eyes have it. *Formulaic sequences: Acquisition, processing, and use 9*, 153.

Van Hee, C., E. Lefever, and V. Hoste (2018a, June). Semeval-2018 task 3: Irony detection in English tweets. In *Proceedings of the 12th International Workshop on Semantic Evaluation (SemEval-2018)*, New Orleans, LA, USA.

Van Hee, C., E. Lefever, and V. Hoste (2018b). We usually don't like going to the dentist: Using common sense to detect irony on twitter. *Computational Linguistics 44*(4), 793–832.

Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin (2017). Attention is all you need. In *Advances in Neural Information Processing Systems*, pp. 5998–6008.

Velasco, D. G. (2007). Lexical competence and functional discourse grammar. *ALFA: Revista de Linguística 51*(2).

Veličković, P., G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio (2017). Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Vervaeke, J. and J. M. Kennedy (1996). Metaphors in language and thought: Falsification and multiple meanings. *Metaphor and Symbol 11*(4), 273–284.

Villavicencio, A. (2003). Verb-particle constructions and lexical resources. In *Proceedings of the ACL 2003 Workshop on Multiword Expressions: Analysis, Acquisition and Treatment - Volume 18*, MWE '03, Stroudsburg, PA, USA, pp. 57–64. Association for Computational Linguistics.

Vincze, V. (2012). *Semi-compositional noun+ verb constructions: Theoretical questions and computational linguistic analyses*. Ph. D. thesis, szte.

Vulchanova, M., E. Milburn, V. Vulchanov, and G. Baggio (2019, Jun). Boon or burden? the role of compositional meaning in figurative language processing and acquisition. *Journal of Logic, Language and Information 28*(2), 359–387.

Wallace, B. C., L. Kertz, E. Charniak, et al. (2014). Humans require context to infer ironic intent (so computers probably do, too). In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Volume 2, pp. 512–516.

Waszczuk, J., R. Ehren, R. Stodden, and L. Kallmeyer (2019, August). A neural graph-based approach to verbal MWE identification. In *Proceedings of the Joint Workshop on Multiword Expressions and WordNet (MWE-WN 2019)*, Florence, Italy, pp. 114–124. Association for Computational Linguistics.

Wilks, Y. (1975). A preferential, pattern-seeking, semantics for natural language inference. *Artificial intelligence 6*(1), 53–74.

Wilson, D. and D. Sperber (1992). On verbal irony. *Lingua 87*(1), 53–76.

Wolf, T., L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, and J. Brew (2019). Hug-

gingface's transformers: State-of-the-art natural language processing. *ArXiv abs/1910.03771*.

Wood, M. S. (2015). *Aristotle and the Question of Metaphor*. Ph. D. thesis, Université d'Ottawa/University of Ottawa.

Wu, Y., M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, et al. (2016). Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Yazdani, M., M. Farahmand, and J. Henderson (2015). Learning semantic composition to detect non-compositionality of multiword expressions. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1733–1742.

Appendix A

Features Used in Subtasks A and B

This is the list of all the features used for the subtasks A and B in the experiments in Chapter 3. For subtask B, we additionally used topic modelling features.

Features can be broadly classified into two categories. Sentiment features, which are listed below:

Sentiment features
leftIntensity
rightIntensity
polarityDiff
contrast
POS1
NEG1
NEUTRAL1
POS2
NEG2
NEUTRAL2

and surface-level text features which are extracted from orthographic information and emoticons:

Surface Text features	Surface Text features
allcaps1	annoyed1
censored1	date1
elongated1	emphasis1
happy1	hashtag1
heart1	kiss1
laugh1	money1
number1	percent1
phone1	repeated1
sad1	shocking1
surprise1	time1
tong1	url1
user1	wink1
allcaps2	annoyed2
censored2	date2
elongated2	emphasis2
happy2	hashtag2
heart2	kiss2
laugh2	money2
number2	percent2
phone2	repeated2
sad2	shocking2
surprise2	time2
tong2	url2
user2	wink2